BACKLOG AND YOU

If you submit an article to the MONTHLY on, say, January 1, you usually learn its fate (that is, acceptance or rejection) by about April 1; three months is the average time for the refereeing process. (It can be as little as two weeks and as much as two years.) An article appears almost exactly six months after it is accepted: copy-editing, typesetting, galley proof, and page proof take that long.

If the rate at which articles are submitted is greater than the rate at which they can be published, a backlog develops. With a large backlog it is possible that the time between submission and appearance becomes not nine months but a year, or a year and a half, or two years. Currently the article backlog of the MONTHLY is zero. With normal refereeing velocity, a paper submitted by January 1 will appear by October 1 or (with a little good luck and a little arm twisting) even before.

Zero backlog is good for authors (quick service) and bad for editors (worry); negative backlog is bad for readers (a shrinking magazine). Readers of the MONTHLY are hereby encouraged to become writers of the MONTHLY. Take advantage of the zero backlog by writing up and sending in your high quality exposition; it will appear and be appreciated much more quickly than usual.

P. R. HALMOS, *Editor*

FACTORIZATION AND PRIMALITY TESTS

JOHN D. DIXON

Department of Mathematics and Statistics, Carleton University, Ottawa, Ontario, Canada

1. Introduction. Although unique factorization for integers is sometimes called the "fundamental theorem of arithmetic," it is only occasionally that a student learns anything about the constructive aspects of this theorem beyond the most elementary facts. Yet there are interesting unsolved mathematical and computational problems involved in factorization of integers and tests of primality, and many of the ideas involved are accessible to undergraduate students. As L. E. Dickson's "History of the Theory of Numbers" [**12**] shows, these problems have attracted the interest of some of the great mathematicians of the past such as Fermat, Euler, Legendre and Gauss, as well as numerous less well-known names; and much of what has been done recently has its roots in this early work. The general availability of computers has had a positive influence on this field, leading to the development of algorithms which previously would not have been feasible, and raising new questions of a theoretical nature concerning the complexity of the underlying problems. The study of new methods of factorization also has taken on an "applied" flavor since the proposal of [**50**] of a form of public-key cryptosystem whose security lies in the assumption that some large integers are hard to factor; an efficient factoring method would enable an opponent to break the system.

John Dixon completed his bachelor's degree (1958) and master's degree (1959) in number theory at the University of Melbourne, Australia. He received his Ph.D. (1961) in group theory under the supervision of Hans Schwerdtfeger from McGill University. Since then he has taught at California Institute of Technology, University of New South Wales, and Carleton University. He is the author of three books: *Problems in Group Theory* (1967), *The Structure of Linear Groups* (1971), and *Modular Representations of Finite Groups* (with B. M. Puttaswamaiah, 1976). During the past 15 years he has had an increasing interest in algorithmic problems in algebra and number theory. His mathematical heroes are the generalists such as Gauss and Hilbert.

In the survey which follows my aim is to describe the problems and the progress which has been made. Most of the survey will refer to work published in the last 10 years, but the reader should be aware that the history of the area goes back much further and that, frequently, methods have been rediscovered in slightly variant forms. In some cases it is difficult to be sure to what extent an idea is new or how much it is implicit in earlier (sometimes much earlier) work.

2. The Ring $\mathbf{Z}_n$. Let n be the integer in whose factorization we are interested. To avoid uninteresting special cases we shall always assume that n is odd and greater than 1, and suppose that its canonical prime factorization is

$$(1) \qquad n = \prod_{i=1}^{s} r_i^{k_i} \qquad (r_i \text{ distinct primes, } k_i \geqslant 1).$$

(In what follows, p, q and r, with or without subscripts, will always designate primes.) We now recall some basic number theoretic facts.

Let $\mathbf{Z}_n$ denote the ring of integers modulo n, and let $x \mapsto x \bmod n$ be the canonical homomorphism of $\mathbf{Z}$ onto $\mathbf{Z}_n$. If d divides n (notation $d|n$), then there is a natural ring homomorphism of $\mathbf{Z}_n$ onto $\mathbf{Z}_d$ which (with abuse of notation) we also write $x \mapsto x \bmod d$. The ring homomorphism

$$(2) \qquad \mathbf{Z}_n \to \prod_{i=1}^{s} \mathbf{Z}_{n_i} \qquad (\text{direct product of rings})$$

given by $x \mapsto (x \bmod n_i)_i$, where $n_i = r_i^{k_i}$ for $i = 1,\ldots,s$, clearly has kernel 0. By counting the number of elements on the two sides we see that the mapping is also surjective, and so (2) is an isomorphism. This is a version of the *Chinese Remainder Theorem* which is often stated in the form: If $m_1,\ldots,m_t$ are integers which are relatively prime in pairs, and their product is m, then for all integers $x_1,\ldots,x_t$ there is a unique integer $x \in [0, m-1]$ such that $x \equiv x_i (\bmod m_i)$ for each i. From the computational point of view it is important to know how to construct x from the values of the x_i. This can be done using the Euclidean algorithm, and an efficient method is described in [**21**, §4.3.2].

The units (elements with multiplicative inverses) in $\mathbf{Z}_n$ are exactly the elements $x \bmod n$ for which the greatest common divisor GCD(n, x) equals 1. The units form a multiplicative group which we shall denote by U_n, and its order is $\phi(n)$, the Euler phi-function. From the isomorphism (2) we obtain the group isomorphism

$$(3) \qquad U_n \simeq \prod_{i=1}^{s} U_{n_i}.$$

(Note: $\phi(n) = \prod_i \phi(n_i) = n\prod_i (1 - (1/r_i))$.)

For an odd prime r it can be proved that for each $k \geqslant 1$ the group U_{r^k} is cyclic. An integer x for which $x \bmod r^k$ is a generator of this group is called a *primitive root modulo* r^k (see, for example, [**60**, Chap. VI]). Since we are ass uming that n is odd, (3) shows that U_n can be generated by a set of s elements.

Exercise 1. Show that U_n cannot be generated by fewer than s elements.

The following elementary property that holds for groups in general will be used repeatedly in what follows. Suppose that x is an element of finite order h in a group G, and that m is a positive integer. Then:

(4) If $x^m = 1$ and $x^{m/q} \neq 1$ for some prime $q|m$, then the largest power of q dividing m also divides h; moreover, if $x^{m/q} \neq 1$ for *all* primes $q|m$, then $m = h$.

Finally, we recall some basic facts about quadratic residues. Let a be an integer and p be a prime with $p \nmid a$. Then a is a *quadratic residue* for p if for some integer c, $c^2 \equiv a (\bmod p)$; otherwise a is a *quadratic nonresidue*. This is summarized by the *Legendre symbol* $\left(\frac{a}{p}\right)$ which (for

$p \nmid a$) takes the values 1 or -1 depending on whether a is a quadratic residue or not. Euler proved that $\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} (\bmod\, p)$. It is useful to introduce the more general *Jacobi symbol* $\left(\frac{a}{n}\right)$. This is defined for all relatively prime a and n with n odd and greater than 1. If n has the prime factorization (1), then $\left(\frac{a}{n}\right)$ is defined in terms of the Legendre symbol as the product $\Pi_i\left(\frac{a}{r_i}\right)^{k_i}$. Again, the Jacobi symbol takes the values 1 and -1, and it is readily seen that $\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right)$ whenever $a \equiv b (\bmod\, n)$, and that $\left(\frac{ac}{n}\right) = \left(\frac{a}{n}\right)\left(\frac{c}{n}\right)$ whenever $\mathrm{GCD}(ac, n) = 1$. Note, however, that $\left(\frac{a}{n}\right) = 1$ does not in general imply that $a \bmod n$ is a square. The important theorem which is basic to many results on quadratic residues is the *Quadratic Reciprocity Law*. It states that if a and n are both odd and relatively prime, then:

$$\left(\frac{-1}{n}\right) = (-1)^{(n-1)/2}, \qquad \left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8},$$

and

$$\left(\frac{a}{n}\right) = (-1)^{(a-1)(n-1)/4}\left(\frac{n}{a}\right).$$

Proofs may be found, for example, in [**60**, Chap. V].

3. Basic Computational Operations. We shall want to analyze the time that it will take to execute various algorithms. The internal hardware of most computers will perform the basic arithmetic operations on integers of some fixed size (single precision), frequently up to about 10^{10}, but operations on larger integers must be handled by a multiprecision package (perhaps written by the programmer). An extensive discussion of such a package is given in [**21**, §4.3.1]. We only observe here that multiprecision operations for integers about the size of n will be slower than single precision operations by factors of up to $c(\log n)^2$ for some constant c. As usual we express this by saying that these multiprecision operations require $O(\log n)^2$ single precision operations.* Among these operations we include addition, subtraction, multiplication and integer division with quotient and remainder, so ring operations in $\mathbf{Z}_n$ will take a comparable time. It is perhaps worthwhile noting that the time taken by a computer to carry out a single precision operation may vary from around 10^{-3} seconds for a small microcomputer, through about 10^{-6} seconds of central processing time for a medium-sized computer, to perhaps 10^{-10} seconds for the fastest computers.

As well as these basic arithmetic operations there are two other important basic number theoretic algorithms (see [**21**] and [**28**]). The first of these is the *Extended Euclidean Algorithm* which, for integers a and b not both 0, computes $d = \mathrm{GCD}(a, b)$ together with integers u and v such that $au + bv = d$. In particular, this permits us to compute the multiplicative inverse of a unit in $\mathbf{Z}_n$. If a and b are bounded in size by n, then the number of multiprecision operations on numbers of this size which are required to execute this algorithm is bounded by a multiple of $\log n$ (see [**21**, §4.5.2]). Almost the same method allows us to compute the Jacobi symbol via the reciprocity law.

The second important algorithm is the *Power Algorithm*. It is useful to describe this in a more general situation. Let R be a ring and suppose that we want to compute x^m where $x \in R$ and m is a positive integer. If we write m in terms of its binary expansion,

$$m = m_0 + m_1 2^1 + \cdots + m_t 2^t$$

with each $m_i = 0$ or 1, then we can compute $z = x^m$ using at most $2t$ multiplications as follows.

*A real-valued function g is said to be of order f at infinity (written $g(n) = O(f(n))$, if $|g(n)|$ is bounded by some constant multiple of $f(n)$ as $n \to \infty$.

Compute $x_0 = x$ and $x_i = x_{i-1}^2$ for $i = 1,\ldots,t$; then z is the product of all x_i for which $m_i = 1$ (the calculations can be carried out without intermediate storage of the m_i and x_i). The algorithm is evidently related to the curiosity which is sometimes referred to as the Russian peasant method of multiplication using successive doubling; and according to [**21**, §4.6.3] the power algorithm appears in Hindu manuscripts which predate 200 BC. In particular, this algorithm permits us to compute x^m in $\mathbf{Z}_n$ using $O(\log m)$ multiprecision operations on integers the size of n. The power algorithm is also useful for computations in algebraic number fields, polynomial rings, matrix rings, and (using matrices) for computing high order terms in sequences given by linear recurrence relations with constant coefficients.

4. Probabilistic Algorithms. A (deterministic) algorithm A accepts as input an element u from a set I of inputs and, after some finite time $t(u)$, produces an output $A(u)$ lying in a specified set of possible outputs. A probabilistic algorithm is a generalization of this concept. A *probabilistic algorithm P* accepts an input u from a set I of inputs together with an element ω selected from a set Ω_u according to a specified probability distribution. Then, with probability 1, the algorithm will, after some finite time $t(u, \omega)$, produce an output $P(u, \omega)$; and moreover the average $t(u)$ of the times $t(u, \omega)$ over Ω_u is finite. If the algorithm is to be useful, then the output $P(u, \omega)$ should have some property independent of ω, and the elements of Ω_u should be easy to generate with the correct distribution. Probabilistic algorithms have been used informally for many years, but the first formal description (which is slightly different from the one we have given) seems to have been given by M. O. Rabin in [**47**]. The advantage of a formal definition is that it allows us to carry out a rigorous analysis of such algorithms. Note that for a fixed input u, and given $\lambda > 1$, the probability that $t(u, \omega)$ exceeds $\lambda t(u)$ is less than $1/\lambda$; hence an estimate of $t(u)$ gives a good idea of how long the algorithm P is likely to run. For almost all practical purposes (excluding certain real-time situations where a strictly bounded running time is vital), a probabilistic algorithm is as satisfactory as a deterministic algorithm with a comparable running time, and frequently the former is more amenable to analysis.

In the situations which we consider below Ω_u will be the set of all sequences of integers from a fixed finite interval. Although ω is an infinite sequence, only the first few terms are computed, each term being selected independently and uniformly at random from a given finite set (this defines the probability distribution on Ω_u). Actually, in practice, we only approximate this procedure since we are likely to use a pseudo-random number generator rather than make truly random choices (see [**21**, Chap. 3]).

EXAMPLE. No deterministic algorithm for finding a quadratic nonresidue for a prime p is known to have a running time bounded by a power of $\log p$ (compare [**9**], but see Lemma 2 below). However, it is easy to *recognize* when an integer a is a quadratic nonresidue by either computing the Legendre symbol using the quadratic reciprocity law or using Euler's criterion ($a^{(p-1)/2} \equiv -1 (\mathrm{mod}\ p)$). Since, for $p > 2$, half of the integers in $[1, p-1]$ are nonresidues, we can find a quadratic nonresidue for p by a probabilistic algorithm simply by trying successive values of a chosen independently and uniformly at random from $[1, p-1]$ until one satisfies the condition. It is easy to see that on the average only two values of a will be tried, so the average number of single precision operations needed to find a quadratic nonresidue of p is $O(\log p)^3$. Note that the output (the nonresidue) depends on the particular random sequence.

REMARK. We should probably not use this algorithm in practice since for many special classes of primes there are easier ways to find quadratic nonresidues. For example, the quadratic reciprocity law shows that we can always take $a = -1$ if $p \equiv 3\ (\mathrm{mod}\, 4)$, $a = 2$ if $p \equiv 5\ (\mathrm{mod}\, 8)$, and $a = 3$ if $p \equiv 17\ (\mathrm{mod}\, 24)$. This covers all odd primes $p \not\equiv 1\ (\mathrm{mod}\, 24)$, and the rules can be extended in an obvious way.

In the problems of factorization and primality testing the size of the integer n is measured by $\log n$ which is roughly proportional to the number of digits of n. One of the principal theoretical questions is whether there are *polynomial-time* algorithms to solve either of these problems. In other words, are there (perhaps probabilistic) algorithms which for each odd integer $n > 1$ can (i) find a proper factor of n if n is composite, or (ii) find a proof that n is prime if n is prime, such that the execution times are bounded by some power of $\log n$. As we see below, these questions are open problems. The evidence that we have suggests that there may be a polynomial-time algorithm for (ii), but perhaps not one for (i). At the same time there are the practical problems of finding and implementing algorithms to solve these problems for general values of n up to a certain size. At the time of writing, composite numbers of around 50 digits are routinely factored and proofs of primality found for primes up to 200 digits.

Exercise 2. Let p and q be primes and suppose that $q - 1 = p^t m$ where $t \geqslant 1$ and $p \nmid m$. Describe a probabilistic algorithm to find an element $b \in U_q$ of order p^t. Use this to give an efficient algorithm which, for each $a \in U_q$, decides whether $x^p = a$ has a solution $x \in U_q$ and if so finds such an x. [Hint: a is a pth power in U_q if and only if a^m has order dividing p^{t-1}. Show that in the latter case x can be written in the form $a^{m^i}b^j$ (compare with [**28**, p. 133] and [**56**]).]

Exercise 3. If g is a primitive root modulo p for an odd prime p, and $g^{p-1} \not\equiv 1 \pmod{p^2}$, show that g is also a primitive root modulo p^k for all $k > 1$. Note that at least one of g^{p-1} and $(g + p)^{p-1}$ is not congruent to 1 modulo p^2, and so either g or $g + p$, is a primitive root modulo p^k.

Exercise 4. Suppose that p is a prime and that $p - 1$ can be completely factored. Describe an efficient probabilistic algorithm to find a primitive root modulo p^k for all $k \geqslant 1$.

5. Certificates of Primality.

A classical algorithm which both tests n for primality and produces a proper factor of n if n is composite is based on the fact that either n is prime or it has a prime factor $r \leqslant \sqrt{n}$. However, the prime number theorem shows that the number of primes less than $\sqrt{n}$ tends asymptotically to $2\sqrt{n}/\log n$, so it is out of the question to check this criterion directly once n becomes reasonably large. Indeed, you might like to consider the following problem (using whatever computing resources you have available). It was posed in a well-known puzzle book [**15**] published in 1907.

Exercise 5. Is the number $n = 111\ldots1$ (19 ones) prime?

As far back as 1877 E. Lucas and T. Pepin showed, in special cases, how we can prove primality indirectly (see [**12**, p. 376]). The basic criterion follows from (4) and the fact that U_n is cyclic of order $n - 1$ if and only if n is prime. In this simple form the criterion is useful only when $n - 1$ can be completely factored:

(5) n is prime if and only if for some integer a, $a^{n-1} \equiv 1 \pmod{n}$ and $a^{(n-1)/q} \not\equiv 1 \pmod{n}$ for each prime $q | n - 1$.

Exercise 6 (T. Pepin 1877). Let n be the Fermat number $F_k = 2^{2^k} + 1$ $(k \geqslant 2)$. If n is prime show that 5 is a quadratic nonresidue for n, and hence (in this case!) that 5 is a primitive root. Deduce that n is a prime if and only if $5^{(n-1)/2} \equiv -1 \pmod{n}$.

Exercise 7. Show that an odd integer $n > 1$ is prime if and only if for each prime $q | n - 1$ there exists an integer a_q such that $a_q^{n-1} \equiv 1 \pmod{n}$ but $a_q^{(n-1)/q} \not\equiv 1 \pmod{n}$.

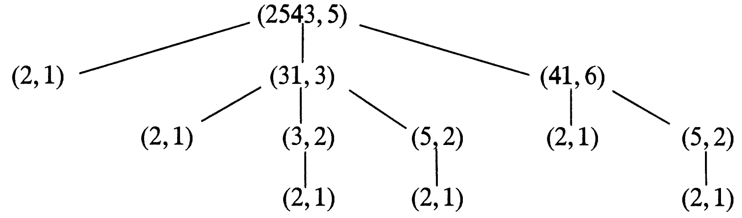
Exercise 8 (H. C. Pocklington 1914). Suppose that $n - 1 = ml$, and that for each prime $q | m$ there exists an integer a_q such that $a_q^{n-1} \equiv 1 \pmod{n}$ and $\mathrm{GCD}(a_q^{(n-1)/q} - 1, n) = 1$. Show that each prime $r | n$ satisfies $r \equiv 1 \pmod{m}$. Thus, if $m \geqslant l$, then n is prime.

Exercise 9 (Analogue of (5)). Let $f(X)$ be a monic polynomial over a field $\mathbf{Z}_p$. Show that $f(X)$ is a product of distinct monic irreducible polynomials in $\mathbf{Z}_p[X]$ all of whose degrees divide a given positive integer d if and only if $f(X) | X^{p^d} - X$. Hence show that a monic polynomial $f(X)$ of degree d is irreducible if and only if $X^{p^d} \equiv X \pmod{f(X)}$ but $X^{p^{d/q}} \not\equiv X \pmod{f(X)}$ for each prime $q|d$. (The latter criterion may be checked in $O(d\log p)^3$ single precision operations using the power algorithm in the ring $\mathbf{Z}_p[X]/(f(X))$. Since approximately $1/d$ of the monic polynomials of degree d in $\mathbf{Z}_p[X]$ are irreducible, the criterion can be used in a probabilistic algorithm to construct an irreducible polynomial of specified degree over $\mathbf{Z}_p$. See also **[10]**, **[19]** and **[32]**.)

If n is composite, then a short proof of this may be given by simply writing n as a product of two proper factors and checking the multiplication. E. T. Bell **[5]** recounts an anecdote in which F. N. Cole did precisely this for the Mersenne number $M_{67} = 2^{67} - 1$ at a meeting of the American Mathematical Society in 1903. Such a proof may be verified by a computation involving $O(\log n)^2$ single precision operations. The catch, of course, is to find the proof—a common difficulty in mathematics.

In the case that n is prime, it is not so obvious that a similarly short proof of primality exists, although proofs in special cases were already given in the last century. However, in 1975 V. P. Pratt **[46]** observed that (5) implies that "every prime has a succinct certificate." Again this proof may be difficult to find, but once written down its validity can be checked easily (Pratt showed that it can be checked in $O(\log n)^5$ single precision operations). The proof can be written as a finite tree whose vertices are labeled by pairs (p, g_p), where p is a prime and g_p is a primitive root modulo p. The root of the tree is labeled (n, g_n), and each vertex labeled (p, g_p) with $p > 2$ has as its children vertices with labels (q, g_q) as q ranges over the primes dividing $p - 1$. The leaves of the tree are all labeled $(2, 1)$.

EXAMPLE*. A certificate of primality for $n = 2543$ is given by



The proof should be checked as follows. For each vertex (k, g_k) with $k > 2$, we should have: (i) $g_k^{k-1} \equiv 1 \pmod{k}$, (ii) $g_k^{(k-1)/l} \not\equiv 1 \pmod{k}$ whenever (l, g_l) is the label of a child of the vertex (k, g_k), and (iii) k is the product of the l (taken to suitable powers) as (l, g_l) runs over the labels of the children of (k, g_k). If these conditions hold, then starting from the leaves $(2, 1)$ the criterion (5) shows successively that the label of every vertex consists of a prime together with an associated primitive root. In particular, n is prime.

R. P. Brent **[7]** has written down proof trees like this for prime factors of the Fermat numbers F_6, F_7 and F_8, but it is not easy to find such proofs since it involves complete factorizations of $p - 1$ for all the primes appearing in the tree. It has been pointed out in **[37]** that, on the other hand, it is rather easy to construct artificially very large primes together with their proof trees by starting out at the leaves of the tree and working up to the root.

The question remains: how do you *find* a proof of primality? Until quite recently, this seemed only a little easier than the factorization problem. Results such as Exercise 8 show that a complete factorization of $n - 1$ can be avoided. E. Lucas showed that in some cases it is possible to use a factorization of $n + 1$ in place of $n - 1$. His idea was generalized by D. H. Lehmer (see **[28**, p. 128]), and more recently it has been shown that partial factorizations of $n^2 \pm 1$ and $n^2 \pm n + 1$ can all be combined to prove primality of n. An extensive survey of these methods is given by H. C. Williams in **[61]**. It now appears, however, that except for special classes of numbers such as the Mersenne numbers, the recent algorithm of L. M. Adleman, R. S. Rumely and C. Pomerance

*As is customary, we have drawn our tree "upside down" with its leaves at the bottom.

is superior on both theoretical and practical grounds for finding proofs of primality. We shall deal with this algorithm in Section 13.

Note: As a curiosity we note that, as at time of writing, the largest known prime is the 39,751-digit Mersenne number $M_{132049} = 2^{132049} - 1$. It was proved prime by David Slowinski in September 1983, using the fact that in this case $n + 1$ is a power of 2.

6. Pseudoprimes and Proofs of Compositeness. To many people it comes as a surprise that it is often easy to show that an integer is composite without having any idea what proper factors it might have. If n is prime, then $|U_n| = n - 1$, and so (as Fermat knew)

$$(6) \qquad b^{n-1} \equiv 1 \pmod{n}$$

holds whenever n is prime and $n \nmid b$. In general, if (6) holds, then we say that n passes the *pseudoprime test to base* b; if, in addition, n is composite, then we call n a *pseudoprime to base* b. (Some people say when (6) holds that n is a "probable prime to base b.") Thus, if for some $b \in [1, n - 1]$, n does not pass the pseudoprime test to base b, then we have a proof that n is composite. This is a simple and often effective way to prove compositeness.

Historically, there was some confusion about the converse: if (6) holds for some b and n, can we deduce that n is prime? It is reported in [**12**, p. 91] that at one stage Leibniz believed that any integer n passing the pseudoprime test to base 2 is prime, and perhaps the fact that each Fermat number $F_k = 2^{2^k} + 1$ passes the pseudoprime test to base 2 misled Fermat into believing that all these numbers are prime (actually numerous Fermat numbers are now known to be composite, but no prime beyond F_4 has been found). The fact is that for each $b > 1$ there are n which are pseudoprime to base b, and it is even true that for some n, n is pseudoprime to base b for all integers b relatively prime to n. On the other hand, for a fixed $b > 1$, the number of pseudoprimes to base b is very small compared with the number of primes (see [**16**] and [**42**]).

Exercise 10. Show that F_5 is composite by proving that it does not pass the pseudoprime test to base 3.

Exercise 11 (R. D. Carmichael 1912). Suppose that n is composite with the factorization (1). Show that (6) holds for all b relatively prime to n if and only if for each i we have $k_i = 1$ and $r_i - 1 | n - 1$. (The latter condition implies that $s \geqslant 3$.) Find examples of such "Carmichael numbers." (D. Shanks notes that if $p > 3$, $2p - 1$ and $3p - 2$ are all primes, then their product is a Carmichael number.)

The existence of Carmichael numbers shows that in some cases it will be difficult to find proofs of compositeness using a pseudoprime test of the type (6). A slightly stronger form of the test does much better. Write $n - 1 = 2^h m$ with m odd. If n is prime, then U_n is cyclic and so has a unique subgroup of order 2 which is generated by -1. Also, for each $x \in U_n$, the order of x divides $n - 1$ and so the order of x^m divides 2^h; hence either $x^m = 1$ or, for some $i \in [0, h - 1]$, x^{m2^i} has order 2 and so equals -1. Thus the condition

$$(7) \qquad \text{either } b^m \equiv 1 \pmod{n} \text{ or } b^{m2^i} \equiv -1 \pmod{n} \text{ for some } i \in [0, h - 1]$$

holds whenever n is prime and $n \nmid b$. Clearly (7) implies (6), and if (7) holds then we shall say that n passes the *strong pseudoprime test to base* b; if, in addition, n is composite, then we say n is a *strong pseudoprime to base* b. Note that using the power algorithm to check (7) will take approximately the same amount of work as checking (6), even though the former test is stricter. As we see in the theorem below there are no analogues of the Carmichael numbers for the test (7), and the power of the strong pseudoprime test is illustrated by computations of [**44**] which show that every odd $n < 25 \cdot 10^9$ which satisfies (7) for $b = 2, 3, 5, 7$ and 11 is prime. The proof of Theorem 1 is based on the following result about the structure of U_n.

LEMMA 1. *Let n be an odd integer > 1, and let B consist of all elements $u \in U_n$ such that the cyclic subgroup $\langle u \rangle$ either has odd order or contains -1. If B generates U_n, then n is a prime power.*

Proof. Assume that n has the factorization (1). Then (3) shows that each element of U_n has order dividing the least common multiple

$$\mathrm{LCM}\{\phi(n_i) | i = 1, \ldots, s\} = 2^t l,$$

say, where l is odd. Choose j so that 2^t divides $\phi(n_j)$. We consider the group homomorphism $\psi : U_n \to U_n$ defined by $\psi(x) = x^{2^{t-1}l}$. For each $x \in B$, $\psi(x) \in \langle x \rangle$ and $\psi(x)^2 = 1$; thus, either $\psi(x) = 1$ or $\psi(x)$ is the (unique) element of order 2 in $\langle x \rangle$, namely -1. If B generates U_n, then $\psi(x) = \pm 1$ for all $x \in U_n$. On the other hand, if n were not a prime power, then $s > 1$ in (3), and so U_n contains an element v such that $v \bmod n_j$ has order 2^t in U_{n_j} and $v \bmod n_i = 1$ for all $i \neq j$; evidently $\psi(v) \neq \pm 1$. Thus we conclude that if B generates U_n, then $s = 1$ and n is a prime power as asserted.

THEOREM 1. (a) *Suppose that $n - 1 = 2^h m$ with m odd, and set*

$$B' = \left\{ x \in U_n | x^m = 1 \text{ or } x^{m2^i} = -1 \text{ for some } i \in [0, h-1] \right\}.$$

If B' generates U_n, then n is prime.

(b) (M. O. Rabin [47]). *If n is composite, then* (7) *fails to hold for at least half of the integers* $b \in [1, n-1]$.

Proof. (a) With the notation of Lemma 1 we have $B' \subseteq B$ and $x^{n-1} = 1$ for all $x \in B'$. Thus, if B' generates U_n, then Lemma 1 shows that $n = r^k$ for some prime r, and $x^{n-1} = 1$ for all $x \in U_n$. But then U_n is cyclic of order $r^{k-1}(r-1)$, and so $r^{k-1}(r-1)$ divides $r^k - 1$, which shows that $k = 1$ as required.

(b) Since B' consists of all elements $b \bmod n$ such that $b \in [1, n-1]$ and (7) holds for b, part (a) shows that B' generates a proper subgroup of U_n. Therefore $|B'| \leqslant \frac{1}{2}|U_n|$, and the result follows.

Note: Actually Rabin uses a slightly more complicated, but equivalent, formulation of the condition (7). In [49] he shows that "half" in the theorem can be replaced by "one quarter," and he gives examples of composite n where this is essentially best possible.

The property proved in Theorem 1(b) can be used to give a compositeness test for integers. (In [47] and elsewhere it is referred to as a primality test, but this is misleading.) Fix an integer $k \geqslant 1$ and choose k integers independently and uniformly at random from the interval $[1, n-1]$. If n fails the strong pseudoprime test to any of these bases, then declare n to be composite; otherwise the test is inconclusive, but there is very strong evidence that n is prime. There has been some confusion about what this test indicates when the second of these alternatives holds. As Rabin emphasizes, it is nonsense to say that n is "probably prime" (it is either prime or not), and there is no value of k for which this probabilistic algorithm will ever prove that n is prime. On the other hand, if n is composite, then, independently of n, the probability that the test will show that n is composite is at least $1 - 2^{-k}$ ($1 - 4^{-k}$ on the basis of the stronger result). Thus, if $k = 100$, say, then it is extremely unlikely that the test will fail to show that n is composite. So, if n passes the strong pseudoprime test for 100 randomly chosen bases, then there is an extremely strong presumption that n is prime, but the proof is lacking. Note that each test of (7) requires $O(\log n)^3$ single precision operations.

Independently, at about the same time as [47], a slightly different probabilistic compositeness test was proposed by R. Solovay and V. Strassen [59] (see also the paper [29] of D. H. Lehmer). In this test we compute the value of the Jacobi symbol $(\frac{a}{n})$ for randomly chosen integers $a \in [1, n-1]$ and compare this with the value of $a^{(n-1)/2} \bmod n$; if n is prime, the two values are the same. However, L. Monier [34a] shows that compared to the Rabin test the Solovay-Strassen test is slightly inferior: it requires more computation and is sometimes less effective in detecting a nonprime. More recent papers [1], [24], [44] (and others) describe related tests.

Is it possible to strengthen Theorem 1 in some way to obtain a result which can be used to prove primality? The difficulty seems to be at about the same level as the problem of recognizing when a subset of U_n generates U_n without knowing the prime factors of n—which seems to be a very difficult problem. However there is a conditional solution based on the unproved Extended Riemann Hypothesis (ERH). The ERH is that whenever χ is a character modulo m, then the L-function $L(s, \chi) = \Sigma k^{-s}\chi(k)$ (summed over all positive integers k relatively prime to m) can be extended by analytic continuation to a meromorphic function *without zeros* in the half plane $\operatorname{Re} s > \frac{1}{2}$. The famous Riemann Hypothesis is the special case where $m = 1$ and χ is identically 1. Since the Riemann Hypothesis has resisted efforts to either prove or disprove it for over a century, it is difficult to evaluate results conditional on the ERH. Still, in absence of anything better we have the following result (see [**35**, p. 120] for a proof).

LEMMA 2 (N. C. Ankeny, H. L. Montgomery). *Assuming* ERH *there exists a constant* $C > 0$ *such that, for all integers* $n > 1$ *and each nontrivial group homomorphism* $\psi: U_n \to G$ *into some group* G, *there exists a prime* q *such that* $1 < q \leqslant C(\log n)^2$ *and* $\psi(q \bmod n) \neq 1$.

As a simple corollary we get useful information about a generating set for U_n.

LEMMA 3. *Assuming* ERH *and taking* C *as the constant in Lemma* 2, *the group* U_n *is generated by the set* B *consisting of the elements* $q \bmod n$ *as* q *ranges over the primes in* $[1, C(\log n)^2]$.

Proof. Let S be the subgroup of U_n generated by B and put $G = U_n/S$. Then the canonical homomorphism $\psi: U_n \to G$ is surjective and has the kernel S. Since $\psi(q \bmod n) = 1$ for all $q \in B$, Lemma 2 shows that ψ is trivial. Hence $G = 1$, and so $S = U_n$.

Together Lemma 3 and Theorem 1(a) imply at once the following conditional result.

THEOREM 2 (G. L. Miller). *Assuming* ERH *and taking* C *as the constant in Lemma* 2, *each odd integer* $n > 1$ *which passes the strong pseudoprime test to base* q *for each prime* $q \in [1, C(\log n)^2]$ *is prime.*

Evidently Theorem 2 can be used to establish (conditionally on ERH) a polynomial-time primality test, and this was done by Miller in [**34**]. Miller's original result is slightly different from our Theorem 2, but it was his work which led to the fruitful idea of a strong pseudoprime test and Rabin's compositeness test.

Exercise 12. Let m and k be positive integers. Show that if k integers $b_1, \ldots, b_k$ are chosen independently and uniformly at random from $[1, m]$, then the probability that at least one composite odd integer $n \in [1, m]$ is a pseudoprime to all the bases $b_1, \ldots, b_k$ is less than $m2^{-k-1}$. In particular, there is a set $T \subseteq [1, m]$ consisting of at most $\log m/\log 2$ integers such that every odd composite $n \in [1, m]$ fails (7) for at least one $b \in T$. (Compare with [**2**]. Unfortunately, this result gives no idea how to find such a set T.)

7. Factorization. In practice, if we want to factor n into its prime factors, then we begin by removing its small prime factors (up to perhaps 10^4) by trial division. We then make one or more pseudoprime tests (7) on the unfactored part until we either discover that it is composite or are strongly convinced that it must be prime. In the latter case we must look for a proof of primality, and that process we shall describe in Section 13. In the former case we must search for a proper factor; when we have found one we have reduced the problem into two simpler problems and can repeat the process on the smaller numbers. At present, the most powerful known methods of factoring routinely succeed in factoring integers of 50 digits in a few hours, but factoring larger integers involves some measure of luck (see [**45**]). Thus successful factorizations of large integers have been achieved by a set of rather ad hoc methods, some of which may be more or less appropriate when faced with a specific integer. In Sections 8–11 we give a summary of the principal methods which have had some success in practice; further details may be found in [**20**] and [**21**]. In Section 12 we discuss some theoretical results.

8. Direct Methods and Sieves. Once we have eliminated small prime divisors, the straightforward method of trial division is usually unfruitful. A sieve method is a refinement which eliminates various classes of possible divisors. For example, suppose n is a Fermat number $F_k = 2^{2^k} + 1$ such that the prime $r|n$. Then the element $x = 2 \bmod r$ in U_r satisfies $x^{2^k} = -1$, and so x has order 2^{k+1} by (4). Hence $2^{k+1}|r - 1$; that is, all prime divisors r of n lie in the arithmetic progression $2^{k+1}m + 1$ $(m = 1, 2, \ldots)$.

Exercise 13. (a) Show that if $k \geqslant 2$ and the prime $r|F_k$, then, in fact, $2^{k+2}|r - 1$. [Hint: $r \equiv 1 \pmod 8$ and so $(\frac{2}{r}) = 1$.]

(b) (Euler 1747). Find a prime factor of F_5.

Clearly the method just described depends very heavily on the special form of n. A more generally applicable sieve method uses quadratic residues. In 1798 Legendre noted that if for some integers a and c relatively prime to n we have $c^2 \equiv a \pmod n$, then for each prime $r|n$ the same congruence holds and so $(\frac{a}{r}) = 1$. If a is odd, then the quadratic reciprocity law shows that $(\frac{r}{|a|})$ can be computed, and this implies that r must lie in certain arithmetic progressions with common difference $|a|$ if $4|a - 1$ or $4|a|$ otherwise. Knowledge of a number of small squares modulo n can lead to useful restrictions on the possible values of r. (See [**23**]. A similar method appears in [**17**, §322].)

Example. As a rather artificial example consider $n = 1711$. Note that $16^2 \cdot 7 - 9^2 = 1711$ and that $37^2 \cdot 5 - 1 = 4 \cdot 1711$. Thus 7 and 5 are both squares modulo n, and so for each prime $r|n$ we have $(\frac{r}{5}) = (\frac{5}{r}) = 1$ and

$$\left(\frac{r}{7}\right) = \left(\frac{7}{r}\right)(-1)^{(r-1)/2} = (-1)^{(r-1)/2}.$$

Thus r is congruent to $\pm 1 \pmod 5$ and to ± 1, ± 9 or $\pm 25 \pmod{28}$. The smallest integers which satisfy these two conditions are $1, 9, 19, 29, \ldots$ and indeed $r = 29$ divides n.

The example shows the two difficulties which arise when this method is applied. Firstly, we need to be able to find relatively small integers which are squares modulo n, and secondly, we need an efficient way of combining the information on r obtained from the different moduli. Legendre suggested using the continued fraction approximations to $(kn)^{1/2}$ for various integers k to solve the first problem (see Section 11 below), and over the past 50 years D. H. Lehmer and his associates have constructed a series of special purpose computers ("delay line sieves") which handle the second problem (see [**27**] and [**30**]). Asymptotically, the method is poor; its running time grows like $cn^{1/2}$ although the constant c may be rather small. However, in the past it has been competitive because the delay line sieves were much faster (for their specific task) than the contemporary general purpose computers. The SRS-181 built in Berkeley in the early 1970s processes $2 \cdot 10^7$ integers per second, and faster sieves have been built since then (for example, by H. C. Williams in Winnipeg).

9. Pollard's Monte Carlo Method (or Rho Method). Let S be a finite set and consider the semigroup Map(S) of all functions of S into itself under the operation of composition. If $\psi \in \mathrm{Map}(S)$ and $x_0 \in S$, then (since S is finite) there exists an integer $l \geqslant 1$ such that the elements $x_0 = \psi^0(x_0), \psi(x_0), \ldots, \psi^{l-1}(x_0)$ are all distinct, but $\psi^l(x_0) = \psi^{l'}(x_0)$ for some $l' \in [0, l - 1]$. We shall call l the *length of the orbit* of x_0 under ψ, and note that once $k \geqslant l'$ the sequence $\{\psi^k(x_0)\}$ is periodic with period $l - l'$.

Now suppose S has m elements and $x_0 \in S$. The length of the orbit of x_0 under ψ is at least l provided $x_0, \psi(x_0), \ldots, \psi^{l-1}(x_0)$ are all distinct. Thus the number of $\psi \in \mathrm{Map}(S)$ for which x_0 has an orbit of length at least l is

$$m^{m-l}\prod_{i=0}^{l-1}(m - i) < m^m \exp\left(-\sum_{i=0}^{l-1} i/m\right) = m^m \exp(-l(l - 1)/2m).$$

Thus, for each $\lambda > 0$, the proportion of $\psi \in \mathrm{Map}(S)$ under which x_0 has an orbit of length at least $(2\lambda m)^{1/2} + 1$ is less than $\exp(-\lambda)$. Hence there will be relatively very few pairs x_0, ψ where the orbit is greater than, say, $5m^{1/2}$; we shall call such pairs "exceptional" (compare [**21**, Ex. 3.1.12]).

J. M. Pollard in [**39**] and [**40**] applied this observation to obtain the following factorization method. Suppose that n is composite and the prime $r|n$. Set $S = \mathbf{Z}_r$, let y_0 be an integer and ψ be a polynomial function. If the pair $y_0 \bmod r$, ψ is not exceptional, then the length of the orbit of $y_0 \bmod r$ under ψ is bounded by a small multiple of $r^{1/2}$. We do not know r so we cannot compute the elements $x_k = \psi^k(y_0 \bmod r)$ in $\mathbf{Z}_r$, but we can compute the sequence $\{y_k\}$ where, for each $k > 0, y_k \in [0, n-1]$ and $y_k \equiv \psi(y_{k-1}) \pmod{n}$; and the latter satisfies $x_k = y_k \bmod r$ for all k. Now, with the notation above, we know that $\{\psi^k(x_0)\}$ is periodic with period $d = l - l'$ once $k \geqslant l'$. This implies that r divides $\mathrm{GCD}(y_k - y_h, n)$ whenever $k > h > l'$ and $d|k-h$. This is a special case of finding cycles in sequences of iterates. It was first solved by R. W. Floyd who noted that we can look for solutions with $k = 2h$; and later R. P. Brent [**6**] showed that it is often better to look for solutions where k and h have the forms $h = 2^i - 1$ and $k = 2^i + j$ with $0 \leqslant j < 2^i$. The point is that by searching for suitable y_h and y_k in this way it is only necessary to retain in memory two values of the sequence, so Pollard's method requires very little storage. Unless we are very unlucky, the result of this search will be a proper factor of n in the form $\mathrm{GCD}(y_k - y_h, n)$.

Example. Let $n = 30623$ and take $\psi(x) = x^2 + 1$ and $y_0 = 1$. Then the successive values of y_k such that $y_k \equiv \psi(y_{k-1}) \pmod{n}$ for all $k > 0$ are: $1, 2, 5, 26, 677, 29608, 19667, 22400, \ldots$. In practice, as k ranges over the interval $[2^i, 2^{i+1} - 1]$ only the current value of y_k and the value of y_{2^i-1} are stored. We find that $y_7 - y_3 = 22374$ has a common divisor of 113 with n, and hence 113 is a proper division of n.

How should we choose ψ and x_0 so that they form a nonexceptional pair for $\mathbf{Z}_r$? Nothing positive seems to be known, in general, although it can be shown that ψ should not be linear, nor of the form $x^2 - 2$. Numerical evidence for the primes up to 10^6 (see [**20**]) indicates that $\psi(x) = x^2 + 1$ or $x^2 - 1$ are, as well as simple to compute, quite good; and these functions are most generally used, regularly producing prime factors in $O(r^{1/2})$ steps. The most notable success of this method was to factor the 78-digit Fermat number F_8 whose compositeness had been proved by J. C. Morehead and A. E. Western in 1909 (see [**8**]). For details and implementation of Pollard's method see [**6**], [**8**], [**20**], [**21**] and [**40**].

Since each composite number n has a prime factor $r \leqslant n^{1/2}$, the Monte Carlo method appears to have its execution time bounded by $O(n^{1/4})$, but this conclusion is conditional on the unproved assumption that the particular x_0, ψ chosen are nonexceptional. It is an open problem whether the method (or some modification) is susceptible to a complete analysis.

Exercise 14. Show that, if $x_0 \in S$ and $|S| = m$, then the average length of the orbit of x_0 under ψ as ψ runs over the *bijections* of S into itself is $(m+1)/2$. (This can be used to explain why a linear polynomial should not be used in Pollard's method.)

Exercise 15 (J. M. Pollard). Let p be prime and let g be a primitive root modulo p. Use the ideas above to give an algorithm which, for any integer a with $p \nmid a$, finds the index i for which $g^i \equiv a \pmod{p}$, and is likely to have its execution time bounded by $O(p^{1/2}(\log p)^3)$. (For related results see [3] and [**38**].)

10. Using the Group Structure of U_n to Find Factors of n. If n has s distinct prime factors, then it follows from (3) that the subgroup $T = \{x \in U_n | x^2 = 1\}$ has order 2^s since each U_{n_i} has a unique subgroup of order 2. Conversely, if for some integer c, $c^2 \equiv 1 \pmod{n}$ and $c \not\equiv \pm 1 \pmod{n}$, then $\mathrm{GCD}(c-1, n)$ is a proper factor of n. Thus every element $x \neq \pm 1$ in T yields a proper factor of n, and many factoring methods are based on this observation.

Exercise 16. Suppose that for some integer $d > 1$ you have $d + 1$ distinct roots in $\mathbf{Z}_n$ for a polynomial in $\mathbf{Z}_n[X]$ of degree d. How can you find a proper factor of n?

Now suppose that we know an integer $m > 0$ with the property that $x^m = 1$ for all $x \in U_n$. Write $m = 2^h l$ with l odd. If n is not a prime power, then Lemma 1 shows that for at least half of the elements $x \in U_n$, the cyclic group $\langle x \rangle$ has even order but does not contain -1. In this latter case there is an index $t \in [0, h-1]$ such that $y = x^{l2^t} \neq \pm 1$ but $y^2 = 1$, and so we can obtain a proper factor of n. This leads to a probabilistic algorithm in which a sequence of values of x are chosen independently and uniformly at random from U_n, and the algorithm halts when it finds a value of x which gives a proper factor of n in the manner above. The average number of values of x which must be chosen is at most 2, so the average number of single precision operations required to find a factor of n is $O((\log n)^2 \log m)$. For example, if we know $\phi(n)$, the order of U_n, then $m = \phi(n)$ will factor n very quickly. The choice of $m = n!$ is also valid but then $\log m$ is too large.

In general, it will not be possible to find a suitable value of m which is small enough to make the method above feasible, unless we have further information on the arithmetic structure of n (which is what we are trying to find!) However, variations of this idea have been suggested by D. N. and D. H. Lehmer (unpublished) and by J. M. Pollard [**39**], and have been used with some success. Fix a constant $c > 0$ (around 10^5 in some applications), and let m be the least common multiple of all prime powers less than c. If n has the factorization (1) and $\phi(r_i^{k_i})|m$ for all i, then $x^m = 1$ for all $x \in U_n$ by (3), and so the method above applies. However, we still obtain a factorization if there is at least one $\phi(r_j)|m$, since in this case we always have $x^m \bmod r_j = 1$ and so $\mathrm{GCD}(x^m - 1, n) \neq 1$. Further modifications (see [**20**], [**39**] and [**62**]) extend the utility of this method, but its success clearly depends to a certain amount on a fortunate choice of n.

EXAMPLE. Let $n = 1711$ and take $c = 10$. Then $m = 2^3 \cdot 3^2 \cdot 5 \cdot 7 = 2520$. Choose $x = 2$. Then $x^m = 523 \pmod n$ and $\mathrm{GCD}(522, n) = 29$ which is a proper factor of n.

Exercise 17. If k is relatively prime to the order of U_n, show that $x \mapsto x^k$ is a bijection of U_n onto itself, and that the inverse mapping also has the form $x \mapsto x^{k'}$ for some k'. Explain how to compute k' from k if the prime factors of n are known. Conversely, show that n may be easily factored if we know a single pair of such integers k, k' (of a size comparable to n) with $k > 1$. (Thus computing the inverse function is as difficult as factoring n into primes. These mappings have been used as "trapdoor functions;" see, for example, [**13**].)

11. Factor Bases and the Continued Fraction Method. As noted in the previous section, we can find a proper factor of n whenever we can find $x \in U_n$ such that $x^2 = 1$ and $x \neq \pm 1$; and, conversely, such an element exists provided n is not a prime power. Evidently, it is enough to find integers a and b such that $a^2 \equiv b^2 \pmod n$ and $a \not\equiv \pm b \pmod n$, since again $\mathrm{GCD}(a - b, n)$ is a proper factor of n. In 1643 Fermat factored integers by making a direct search for a and b such that $n = a^2 - b^2$ (see [**12**, p. 357]). More recently R. S. Lehman [**25**] showed that a modification of Fermat's idea using Farey fractions gives a factoring algorithm whose running time is $O(n^{1/3})$, but this does not seem to be competitive with the known alternatives. A better approach is the following.

We define a *factor base* to be a set B of nonzero integers $\{b_0, b_1, \ldots, b_h\}$ (for example, -1 and the first h primes). Call an integer a a *B-number* if the integer c defined by $c \equiv a^2 \pmod n$ and $c \in [-n/2, n/2]$ can be written as a product of factors from B. Thus, if a is a B-number, then we can compute integers $e(a, i) \geqslant 0$ such that $a^2 \equiv \prod_{i=0}^{h} b_i^{e(a,i)} \pmod n$. To each B-number a we associate the $(h+1)$-vector $e(a) = (e(a,0), \ldots, e(a,h))$. If we can find a sufficiently large set A of B-numbers ($|A| = h + 2$ is enough), then the vectors $e(a)$ $(a \in A)$ are linearly dependent when considered modulo 2. So using ordinary linear algebra (over $\mathbf{Z}_2$) we can find a nonempty subset A' of A such that

$$\sum_{a \in A'} e(a) \equiv (0, \ldots, 0) \pmod 2,$$

and then $h_i = \frac{1}{2}\Sigma_{a \in A'} e(a, i)$ is an integer for each i. If we now define

$$u \equiv \prod_{a \in A'} a (\bmod n) \text{ and } v \equiv \prod_{i=0}^{h} b_i^{h_i} (\bmod n),$$

then $u^2 \equiv v^2 (\bmod n)$. If we are lucky, then $u \not\equiv \pm v (\bmod n)$ and so we obtain a proper factor of n.

EXAMPLE. Let $n = 1711$. Then

$41^2 - 1711 = -30, 83^2 - 4 \cdot 1711 = 45, 124^2 - 9 \cdot 1711 = -23$ and $455^2 - 121 \cdot 1711 = -6$. Using the base $B = \{-2, 3, 5\}$ we obtain

$$41^2 \equiv (-2) \cdot 3 \cdot 5, \quad 83^2 \equiv 3^2 \cdot 5 \text{ and } 455^2 \equiv (-2) \cdot 3 (\bmod n).$$

Although we have only three congruences, it turns out that the three vectors of exponents, namely (1 1 1), (0 2 1) and (1 1 0) are linearly dependent modulo 2 since the sum is 2(1 2 1). Thus

$$(41 \cdot 83 \cdot 455)^2 \equiv ((-2) \cdot 3^2 \cdot 5)^2 (\bmod n),$$

but we are unlucky since

$$41 \cdot 83 \cdot 455 \equiv -(-2) \cdot 3^2 \cdot 5 (\bmod n).$$

However, with the additional equation $185^2 - 20 \cdot 1711 = 5$ we have the further exponent vector (0 0 1), and since

$$(0\,2\,1) + (0\,0\,1) = 2(0\,1\,1)$$

we obtain $(83 \cdot 185)^2 \equiv (3 \cdot 5)^2 (\bmod n)$ and from this get the proper factor $\text{GCD}(83 \cdot 185 - 3 \cdot 5, n) = 29$.

This idea can be traced back to M. Kraitchik [**22**] in 1926, but it is not described in any systematic way. In 1931 D. H. Lehmer and R. E. Powers [**31**] suggested a similar idea in which a method of Legendre was used to generate relatively small quadratic residues (which might be expected to factor into products of small primes more often). Their method was impractical until the advent of fast computers with large memory resources. In the late 1960s, using [**31**] as a basis, J. Brillhart and M. A. Morrison systematized and implemented a computer version of the method. Almost at once they had their first major success: they factored the 39-digit Fermat number F_7 which had been known to be composite since 1897 (see [**36**]). The *continued fraction method* as it is now called is presently the most consistently successful method for factoring large numbers up to about 50 digits.

What then was Legendre's idea for finding small quadratic residues? In general, for any irrational number, the continued fraction expansion gives a sequence of increasingly close rational approximations (see, for example, [**60**, Chap. 1]). In particular, if n is not a square, then, using only integer arithmetic, we can compute two increasing integer sequences $\{u_k\}$ and $\{v_k\}$ such that, for each k,

$$|n^{1/2} - u_k/v_k| < 1/v_k v_{k+1},$$

and hence $u_k^2 - n v_k^2 = w_k$, say, with $|w_k| < 2n^{1/2}$. Thus $\{w_k\}$ is a sequence of quadratic residues for n whose sizes are $O(n^{1/2})$. Moreover, n is a quadratic residue for any prime dividing w_k. Taking our factor base B to consist of -1 and the first h primes p for which $(\frac{n}{p}) = 1$ (for a suitable value of h), the relatively small size of the w_k increases the chance that u_k is a B-number, and in this way we may hope to generate enough B-numbers to make the method work. We should note several points which are important in practice. There is no need to compute the integers v_k, and the integers u_k are only computed modulo n; and it is sometimes advantageous to work with the continued fraction for $(dn)^{1/2}$ for some small integer d in place of $n^{1/2}$. A lot of work has been done to ensure that the implementation is efficient as possible, and some of this is described in the original paper [36]. Data given in the paper [63] suggests that in the range 10^{30} to 10^{50} the average running time is roughly proportional to $n^{1/7}$. The recent paper [45] discusses several devices which have had the effect of speeding up the algorithm by a factor of 10–15; one of these is an "early abort strategy" which discards terms w_k at an early stage when there is evidence that they do not have enough small factors. No one has given a complete theoretical analysis of the effectiveness of the continued fraction algorithm, but two recent papers have shown that on plausible (but unproved) assumptions about the distribution of primes, the asymptotic running time is of the form

$$\exp\left\{c(\log n \log\log n)^{1/2}\right\} = n^{c(\log\log n/\log n)^{1/2}}$$

(see [43] and [51]).

12. Theoretical Estimates for the Factoring Problem. It is unfortunately true that none of the factoring methods which are most effective in practice has had an adequate analysis of its performance. As noted above, the continued fraction method is well supported by empirical evidence (see [63] and [45]); [62] gives some statistics on the method discussed in Section 10; [20] reports computational evidence in a limited range which supports the Monte Carlo method; and both [43] and [51] give analyses of the continued fraction method and its variants based on plausible but unproved conjectures about the distribution of primes and prime divisors. Still there is no assurance that for some (perhaps most!) values of n these methods might all work quite badly. It is therefore of interest to know that there is a method which, although undoubtedly poor from a practical point of view, can be completely analysed and shown to have a running time which is $O(n^{\varepsilon})$ for each $\varepsilon > 0$.

This asymptotically fast factoring algorithm is a probabilistic algorithm based on a naive use of a factor base. Using the notation of Section 11 consider the base $B = \{b_0, \dots, b_h\}$ consisting of -1 and the first h primes. The basic step in the algorithm consists of choosing independently and uniformly at random enough integers from $[1, n-1]$ until we have a set A of B-numbers whose exponent vectors $e(a)$ $(a \in A)$ are linearly dependent modulo 2. As was shown in Section 11 this leads to a pair of integers u, v such that $u^2 \equiv v^2 \pmod{n}$, and we obtain a proper factor of n provided $u \not\equiv \pm v \pmod{n}$. It is now not too difficult to prove the following theorem.

THEOREM 3 (J. D. Dixon). *Suppose that n is not a prime power. Then the basic step described above will find a proper factor of n with probability at least* $\frac{1}{2}$. *Moreover, there exist absolute constants* $c_1, c_2 > 0$ *such that if* $h = [\exp\{c_1(\log n \log\log n)^{1/2}\}]$ *and the basic step is repeated until a proper factor has been found, then the average running time is* $O(\exp\{c_2(\log n \log\log n)^{1/2}\})$.

REMARK. In the original paper [14] the values $c_1 = \sqrt{2}$ and $c_2 = 3\sqrt{2}$ were obtained. These were improved in [21] and [51], and in [43] it is shown that the average running time for an optimal choice of h is of the form $\exp\{(2 + \varepsilon_n)(\log n \log\log n)^{1/2}\}$, where $\varepsilon_n \to 0$ as $n \to \infty$. Thus the running time grows subexponentially in $\log n$ but faster than any power of $\log n$.

The papers [43] and [51] carefully analyze a whole family of factoring algorithms which are basically of the type above but which use various ways to try to choose the set A of B-numbers more effectively. These papers make precise some very rough arguments which had been given to support proposed methods. The analyses are not complete (they are based on clearly stated, but unproved, heuristic assumptions), but their value lies in suggesting how various modifications might be expected to affect the performance of the algorithms. As well as dealing with the continued fraction method, both papers include interesting details of factoring methods which we have not discussed here. Among these are the linear sieve of R. Schroeppel (see [43]); the quadratic sieve of Pomerance (see [43] and [18]); and variants of a method of D. Shanks (see [55], [51] and [52]) which works in the group of equivalence classes of binary quadratic forms with discriminant $-n$ (and brings us back to Gauss's original investigations [17]). We also mention another method of Shanks christened SQUFOF which appears quite effective for certain ranges of n (see [58]).

13. Finding Primality Proofs in Almost Polynomial Time. The final stage in a complete factorization of an integer is to prove primality of its factors. Suppose that n is a large integer which we suspect of being prime. We should certainly carry out enough pseudoprime tests of type (7) until we are convinced that there is a strong presumption that n is prime, but at that point we are faced with having to find a proof. Until recently, there was no reasonably fast, uniform method of proceeding, and although it was frequently possible to provide certificates of primality for some primes of 80 digits and above, often even smaller primes caused trouble (see **[61]**). However, in 1980, L. M. Adleman and R. S. Rumely proposed a general algorithm for proving primality, and using work of K. Prachar and P. X. Gallagher, A. Odlyzko and C. Pomerance proved that the algorithm would run in close to polynomial-time, namely $O((\log n)^{c \log\log\log n})$ for a suitable constant $c > 0$ (see **[4]**). H. W. Lenstra, Jr., and H. Cohen have produced a streamlined version of this algorithm and Cohen, A. K. Lenstra and D. T. Winter have carefully programmed it (see **[11]** and **[33]**). The most recent report is that this program can routinely provide proofs of primality for primes up to 200 digits in less than 10 minutes. However, the question of whether there exists an algorithm which finds proofs of primality in strictly polynomial-time remains open.

The basic idea behind the Adleman-Rumely algorithm can be described as follows. Working in cyclotomic fields, we carry out a series of generalized pseudoprime tests on n. If any of these tests fails, then n is not prime; if they succeed, then we use the information obtained from the tests to construct a sieve restricting the possible prime divisors of n (compare with Section 8). If n is a prime, then, because we know a great deal about the arithmetic structure of the rings in which we are working in this case, we can choose the tests so as to produce a very simple sieve. The result is that, if the tests are satisfied, then we can prove that each prime divisor r of n must lie in one of a relatively small set of arithmetic progressions which are of the form $t^i \pmod w$ for $i = 0, 1, \ldots$ where $w > n^{1/2}$. Moreover, the choice of w ensures that each element of U_w has order at most

$$z = O\left((\log n)^{c \log\log\log n}\right);$$

so only values of i less than z need to be considered. Thus, to prove that n is prime (or discover that after all it is not), it only remains to check that n has no proper divisor among this latter set. (In the probabilistic version considered below we may take $t = n$.) A critical part of the algorithm is how it is possible to transfer so much information from the pseudoprime tests over to the sieve. In the original version [4] this was done via a general reciprocity law (the Eisenstein reciprocity theorem is adequate), but in the version we give below we follow **[33]** which replaces the use of a reciprocity law by an elementary argument on Gaussian sums.

Theorem 4 below describes the basic generalized pseudoprime tests to which we have referred. To explain these tests and to prepare for the proof of the theorem we begin with a discussion of some elementary properties of Gaussian sums (see **[20a]**).

Let p and q be primes not dividing n such that $p \mid q - 1$, and let ζ_p and ζ_q be primitive pth and qth roots of unity in **C**. Let $g = g_q$ be a generator of the cyclic group U_q (of order $q - 1$). Since $p \mid q - 1$, there exists a homomorphism χ_{pq} of U_q into the cyclic group $\langle \zeta_p \rangle$ defined by $\chi_{pq}(g^j) = \zeta_p^j$ for all j (χ_{pq} is called a "character of degree p with conductor q"). Now for each integer t we define the *Gaussian sum*

$$\tau\left(\chi_{pq}^t\right) = \sum_u \chi_{pq}(u)^t \zeta_q^u$$

where u runs over U_q (the expression ζ_q^u has the obvious meaning; the value of ζ_q^j depends only on the value of $j \bmod q$ so there is no ambiguity). Note that $\tau(\chi_{pq}^t)$ lies in the ring $R_{pq} = \mathbf{Z}[\zeta_p, \zeta_q]$.

LEMMA 4. *Writing χ for χ_{pq} we have*:

(a) $\tau(\chi)\tau(\chi^{-1}) = \chi(-1)q$;

(b) *if n is prime, then* $\tau(\chi)^{n^{p-1}-1} \equiv \chi(n) \pmod n$ *in* R_{pq};

(c) *if for some prime $r \neq p$ we have* $\zeta_p^i \equiv \zeta_p^j \pmod r$, *then* $\zeta_p^i = \zeta_p^j$.

Proof. In all the sums below the variables range over U_q. Several times we use the fact that if v

is a fixed element of U_q (or an integer relatively prime to q), then uv ranges over U_q as u ranges over U_q.

(a)
$$\tau(\chi)\tau(\chi^{-1}) = \sum_u \sum_v \chi(u)\chi(v)^{-1}\zeta_q^{u+v}$$
$$= \sum_u \sum_v \chi(uv)\chi(v^{-1})\zeta_q^{uv+v}$$
$$= \sum_u \chi(u) \sum_v \zeta_q^{v(u+1)}.$$

Since $\zeta_q^q = 1$,
$$\sum_v \zeta_q^{vw} = -1 + (\zeta_q^{qw} - 1)/(\zeta_q^w - 1) = -1$$
if $q \nmid w$, and the lefthand sum is $q - 1$ otherwise. On the other hand,
$$\sum_u \chi(u) = \sum_u \chi(gu) = \chi(g) \sum_u \chi(u),$$
so $\Sigma_u \chi(u) = 0$ since $\chi(g) \neq 1$. Therefore
$$\tau(\chi)\tau(\chi^{-1}) = -\sum_u \chi(u) + q\chi(-1) = q\chi(-1)$$
as asserted.

(b) To simplify notation we shall temporarily put $m = n^{p-1}$. Since n is prime, we have
$$(\alpha_1 + \cdots \alpha_t)^{n^i} \equiv \alpha_1^{n^i} + \cdots + \alpha_t^{n^i} (\operatorname{mod} n)$$
for all $\alpha_1, \dots, \alpha_t \in R_{pq}$ and $i > 0$. Thus
$$\tau(\chi)^m = \Big(\sum_u \chi(u)\zeta_q^u\Big)^m \equiv \sum_u \chi(u)^m \zeta_q^{um} (\operatorname{mod} n).$$
Now $\chi(u)^m = \chi(u)$ since $p | m - 1$, and there exists $v \in U_q$ such that $vm = 1$ since $q \nmid m$. Thus
$$\sum_u \chi(u)^m \zeta_q^{um} = \sum_u \chi(u)\zeta_q^{um} = \sum_w \chi(vw)\zeta_q^w$$
$$= \chi(v)\tau(\chi) = \chi(n)^{1-p}\tau(\chi) = \chi(n)\tau(\chi)$$
because $q | p - 1$ and $vn^{p-1} = 1$. Finally, since $\chi(-1)q = \pm q$ is relatively prime to n, there exists an integer a such that $\chi(-1)qa \equiv 1$ (mod n). So multiplying the congruence
$$\tau(\chi)^m \equiv \chi(n)\tau(\chi)(\operatorname{mod} n)$$
through by $a\tau(\chi^{-1})$ and using (a) we obtain the desired result.

(c) Set $k = i - j$. Then the hypothesis implies that $\zeta_p^k \equiv 1$ (mod r), and hence
$$f(\zeta_p^k) \equiv f(1) = p (\operatorname{mod} r)$$
where $f(X)$ is the cyclotomic polynomial
$$(X^p - 1)/(X - 1) = X^{p-1} + X^{p-2} + \cdots + 1.$$
Since $r \nmid p$, $f(\zeta_p^k) \neq 0$ and so ζ_p^k is not a primitive pth root of 1. Hence $\zeta_p^k = 1$ and so $\zeta_p^i = \zeta_p^j$. This completes the proof of the lemma.

Now let P and Q be finite sets primes not dividing n where, for all $q \in Q$, $q - 1$ is a product of distinct primes from P. Put $z = \prod_{p \in P} p$ and $w = \prod_{q \in Q} q$. Then $q - 1 | z$ for each $q \in Q$, and so the order of each element in U_w (compare with (3)) must divide z. Thus $a^z \equiv 1$ (mod w) for each integer a which is relatively prime to w.

THEOREM 4 (H. W. Lenstra Jr.). *Suppose that n is an odd integer satisfying the following conditions*:

(i) *For all primes p and q with $q \in Q$ and $p|q-1$, $\tau(\chi_{pq})^{n^{p-1}-1} \equiv \chi_{pq}(n) \pmod{n}$*;

(ii) *For each prime $p \in P$ and each prime $r|n$, $p^{e_p}|r^{p-1}-1$ where p^{e_p} is the largest power of p dividing $n^{p-1}-1$.*

Then each prime $r|n$ is congruent to $n^i \pmod{w}$ for some $i \in [0, z-1]$.

REMARK. If n is prime, then part (b) of Lemma 4 shows that (i) is satisfied, and (ii) is trivially satisfied. There are probably very few other integers which satisfy the conditions.

Proof. Condition (ii) shows that, for each $p \in P$ and each prime $r|n$, there exist integers a_p and $b_p(r)$ such that

$$n^{p-1} = 1 + a_p p^{e_p}, \; r^{p-1} = 1 + b_p(r) p^{e_p}$$

and $p \nmid a_p$. Thus for some integer $l_p(r)$ we have $a_p l_p(r) \equiv b_p(r) \pmod{p}$.

Now, for each pair p, q with $q \in Q$ and $p|q-1$, Lemma 4(b) shows that

$$\chi(r) \equiv \tau(\chi)^{r^{p-1}-1} (\operatorname{mod} r)$$

for each prime $r|n$ (where again we have put χ for χ_{pq}). However,

$$(r^{p-1}-1)a_p = (n^{p-1}-1)b_p(r),$$

and so using the congruence in (i) taken modulo r we conclude that

$$\chi(r)^{a_p} \equiv \chi(n)^{b_p(r)} (\operatorname{mod} r),$$

and so $\chi(r)^{a_p} = \chi(n)^{b_p(r)}$ by Lemma 4(c). Since both sides of this equation are pth roots of 1, and $p \nmid a_p$, we conclude that $\chi(r) = \chi(n)^{l_p(r)}$ by the choice of $l_p(r)$. This holds for each prime $r|n$ and all $p \in P$ and $q \in Q$ with $p|q-1$.

Finally, we use the Chinese Remainder Theorem to find an integer $l(r) \in [0, z-1]$ such that $l(r) \equiv -l_p(r) \pmod{p}$ for each $p \in P$. Then, since $\chi(n)$ is a pth root of 1,

$$\chi(rn^{l(r)}) = \chi(r)\chi(n)^{l(r)} = 1,$$

and so $rn^{l(r)}$ lies in the kernel of $\chi = \chi_{pq}$ for all $q \in Q$ and all $p|q-1$. Since $q-1$ is squarefree, the intersection of these kernels as p runs over the prime divisors of $q-1$ is trivial. Thus $rn^{l(r)} \equiv 1 \pmod{q}$ for all $q \in Q$, and so $rn^{l(r)} \equiv 1 \pmod{w}$. Hence $r \equiv n^i \pmod{w}$ with $i = z - l(r)$, and the theorem is proved.

Exercise 18. Suppose that for some prime q (not necessarily in Q) we have $p|q-1$, $\chi_{pq}(n) \neq 1$ and the congruence in condition (i) of Theorem 4 holds. Show that condition (ii) of Theorem 4 will then hold for p. [Hint: Let h be the order of $\tau(\chi_{pq})$ mod r in the group of units of the ring R_{pq}/rR_{pq}, and show that $p^{e_p+1}|h$. Then use Lemma 4(b) to show that $h|(r^{p-1}-1)p$.]

To apply Theorem 4 we have to be able to verify the hypotheses. One way to do this is to first verify (i) using the power algorithm, and then use Exercise 18. The latter requires a search for a prime $q \equiv 1 \pmod{p}$ with $\chi_{pq}(n) \neq 1$. In practice this is quite fast since (asymptotically) $(p-1)/p$ of all primes q satisfying $q \equiv 1 \pmod{p}$ also satisfy $\chi_{pq}(n) \neq 1$. However, a proof that a small "good" prime q exists seems to require the ERH. An alternative approach given in **[11]** avoids this problem.

The remaining question is how we should make appropriate choices for the sets P and Q. In order that a primality proof based on Theorem 4 should be efficient, we should like P and Q small so that the conditions (i) and (ii) can be quickly verified, but at the same time like z small and w large so that the potential prime divisors can be disposed of quickly. By choosing $w > n^{1/2}$ we ensure that each congruence class $n^i \bmod w$ contains at most one potential prime divisor $r \leqslant n^{1/2}$, and then the final step can be executed in $O(z)$ multiprecision operations on numbers the size of n. Moreover, if z and w are both bounded by n, then $|P|$ and $|Q|$ are both $O(\log n)$ which has the effect that (i) and (ii) may both be verified within a time polynomial in $\log n$. Thus, the following theorem (see [4] for a proof) shows that a suitable choice of P and Q gives an algorithm with running time $O((\log n)^{c \log \log \log n})$ for any $c > c_2$.

THEOREM 5 (C. Pomerance, A. Odlyzko). *Let P, Q, z and w be defined as above such that Q consists of all primes q with $q - 1|z$. Then there are absolute computable constants $c_1, c_2 > 0$ such that, if $w > n^{1/2} \geqslant 10$, then the least possible value of z satisfies*

$$(\log n)^{c_1 \log \log \log n} < z < (\log n)^{c_2 \log \log \log n}.$$

REMARK. Computations in [4] show that if P is taken as the set of the first 6 primes, then $z = 30030$, Q consists of 21 primes and $w > (5 \cdot 10^{89})^{1/2}$. Similarly, if P is the set of the first 13 primes, then z is about $3 \cdot 10^{14}$, Q consists of 807 primes and $w > (10^{11356})^{1/2}$. These are optimal choices in the sense of Theorem 5.

The current implementation [11] differs from the algorithm described above in several ways. Firstly, it uses Jacobi sums (which were introduced in the original paper [4]) in place of Gauss sums. The computations required to verify condition (i) of Theorem 4 can then be replaced by computations in a smaller ring $\mathbf{Z}_n[\zeta_p]$ and the exponent $n^{p-1} - 1$ reduced to one about the size of n. Secondly, a generalization of Theorem 4 allows multiple prime factors in z and w and this has a significant effect on the practical running time, although not on its asymptotic form. The result is an efficient primality proving program based on an algorithm which is a major theoretical and practical advance on any earlier method. Finally, we mention that [4] and [33] also give completely deterministic algorithms for finding proofs of primality, and these algorithms have the same kind of asymptotic running-time as the version described above. However, the probabilistic version turns out to be the better choice for implementation.

14. What of the Future? On the theoretical side there still remain the questions as to whether there exist polynomial-time algorithms for proving primality or for factoring integers. The prospect for a polynomial-time algorithm for proving primality seems fairly good, but it may turn out that, on the contrary, factoring is NP-hard.*

The following are some directions which might lead to better theoretical or practical algorithms.

1. Do there exist "highly composite" integer-valued functions which can be computed quickly? In a sense this is the idea behind the method described in Section 10 where $x^m - 1$ is divisible by every prime r such that $r - 1|m$. For example, if $k! \bmod n$ could be computed quickly for $k \in [1, n]$, then we should be able to factor n quickly (see [54] for a mathematical pun on this theme).

2. Can we speed up the method described in Section 11? For example, are there better ways than the use of continued fractions to produce small quadratic residues modulo n (smaller residues should increase the likelihood of a larger portion being B-numbers). Are there better choices for the factor base B than small primes? See [43] for descriptions of other variants of the basic method of Section 11.

3. The Adleman-Rumely-Pomerance algorithm of Section 13 uses the fact that n passes certain kinds of pseudoprime tests to set up an especially attractive form of sieve. Is an analogous idea possible when n is not prime? Specifically, can we construct a sieve so that the potential divisors which remain lie in a simply describable set?

4. What about the use of rings and groups other than $\mathbf{Z}_n$ and U_n? There is a "$p + 1$ method" of factoring which is based on computations in the ring $\mathbf{Z}_n[X]/(X^2 - c)$ (see [20, p. 73] and [62] where the idea appears in a slightly disguised form in terms of Lucas series). D. Shanks has shown how to use the class groups of binary quadratic forms (see [51], [53], [55], [57] and [58]). As Section 13 suggests, the extra freedom might be very helpful.

5. Can the theoretical bases of the currently used methods of factorization be tidied up?

*The important concepts of NP-hard and NP-complete problems are discussed, for example, in [16a].

Perhaps there are probabilistic versions which might be more amenable to analysis.

The bibliography below lists only the books and papers which have been mentioned in this survey. Further references may be found in [**20**], [**41**], [**43**], [**61**] and the Math. Centrum Tracts in which the articles [**43**] and [**53**] appear.

Acknowledgements. I am indebted to R. P. Brent, D. A. Buell, H. W. Lenstra, Jr., C. Pomerance and C. P. Schnorr, all of whom supplied me with source material and references. Very special thanks are due to Carl Pomerance who also read through an earlier version of the manuscript and offered many helpful corrections and criticisms (but, of course, is not responsible for errors which remain). I also wish to acknowledge the hospitality which I have enjoyed both at the Department of Pure Mathematics, I.A.S., Australian National University and at the Mathematics Institute, University of Warwick, during the time that I prepared this paper.

References

1. W. Adams and D. Shanks, Strong primality tests that are not sufficient, Math. Comp., 39 (1982) 255–300.

2. L. M. Adleman, Two theorems on random polynomial time, Proc. IEEE Symp. Found. Comp. Sci., 19 (1978) 75–83.

3. ______, A subexponential algorithm for the discrete logarithm problem with applications to cryptography, Proc. IEEE Symp. Found. Comp. Sci., 20 (1979) 55–60.

4. L. M. Adleman, C. Pomerance and R. S. Rumely, On distinguishing prime numbers from composite numbers, Ann. of Math., (2) 117 (1983) 173–206.

5. E. T. Bell, Mathematics, Queen and Servant of Science, Bell, London, 1951.

6. R. P. Brent, An improved Monte Carlo factorization algorithm, BIT, 20 (1980) 176–184.

7. ______, Succinct proofs of primality for factors of some Fermat numbers, Math. Comp., 38 (1982) 253–255.

8. R. P. Brent and J. M. Pollard, Factorization of the eighth Fermat number, Math. Comp., 36 (1981) 627–630.

9. D. A. Burgess, On character sums and primitive roots, Proc. London Math. Soc., (3) 12 (1962) 179–192.

10. D. G. Cantor and H. Zassenhaus, A new algorithm for factoring polynomials over finite fields, Math. Comp., 36 (1981) 587–592.

11. H. Cohen and H. W. Lenstra, Jr., Primality testing and Jacobi sums, Math. Comp., 42 (1984) 297–330.

12. L. E. Dickson, History of the Theory of Numbers, vol. 1 (reprint), Chelsea, New York, 1952 (original publication 1919).

13. W. Diffie and M. E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory, 22 (1976) 644–654.

14. J. D. Dixon, Asymptotically fast factorization of integers, Math. Comp. 36 (1981) 255–260.

15. H. E. Dudeney, Canterbury Puzzles (reprint), Dover, New York, 1958 (original first edition published 1907).

16. P. Erdős, On pseudoprimes and Carmichael numbers, Publ. Math. Debrecen, 4 (1956) 201–206.

16a. M. R. Garey and D. S. Johnson, Computers and Intractibility, Freeman, San Francisco, 1979.

17. C. F. Gauss, Disquisitiones Arithmeticae (transl. A. A. Clarke S. J.), Yale, New Haven, 1966 (original first edition 1801).

18. G. L. Gerver, Factoring large numbers with a quadratic sieve, Math. Comp., 41 (1983) 287–294.

19. H. Gunji and D. Arnon, On polynomial factorization over finite fields, Math. Comp., 36 (1981) 281–287.

20. R. K. Guy, How to factor a number, in Proc. 5th Manitoba Conf. on Numerical Math., 1975, pp. 49–89.

20a. K. Ireland and M. Rosen, A Classical Introduction to Modern Number Theory, Springer, New York, 1982.

21. D. E. Knuth, The Art of Computer Programming, vol. 2 (2nd ed.), Addison-Wesley, Reading, Mass., 1981.

22. M. Kraitchik, Théorie des nombres, vol. 2, Gauthier-Villars, Paris, 1926.

23. A. M. Legendre, Théorie des nombres, vol. 1 (3rd ed.), Paris, 1830 (1st ed. published 1798).

24. D. J. Lehman, On primality tests, SIAM J. Comput., 11 (1982) 374–375.

25. R. S. Lehman, Factoring large integers, Math. Comp., 28 (1974) 637–646.

26. D. H. Lehmer, An extended theory of Lucas functions, Ann. of Math. (2) 31 (1930) 419–448.

27. ______, The sieve problem for all-purpose computers, Math. Comp., 7 (1953) 6–14.

28. ______, Computer technology applied to the theory of numbers, in Studies in Number Theory (W. J. LeVeque, ed.), MAA Studies in Math., 6 (1969) pp. 117–151.

29. ______, Strong Carmichael numbers, J. Austral. Math. Soc. Ser. A, 21 (1976) 508–510.

30. ______, A history of the sieve process, in A History of Computing in the Twentieth Century (N. Metropolis, J. Howlett and G. -C. Rota, Editors), Academic Press, New York, 1980, pp. 445–456.

31. D. H. Lehmer and R. E. Powers, On factoring large numbers, Bull. Amer. Math. Soc., 37 (1931) 770–776.

32. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovâsz, Factoring polynomials with rational coefficients, Math. Ann., 261 (1982) 515–534.

33. H. W. Lenstra, Jr., Primality testing algorithms [after Adleman, Rumely and Williams], in Sèminaire Bourbaki, vol. 1980/81, Lecture Notes in Math. 901, Springer, Berlin, 1981.

34. G. L. Miller, Riemann's hypothesis and tests for primality, J. Comput. System. Sci., 13 (1976) 300–317.

34a. L. Monier, Evaluation and comparison of two efficient probabilistic testing algorithms, Theoret. Comp. Sci., 12 (1980) 97–108.

35. H. L. Montgomery, Topics in Multiplicative Number Theory, Lecture Notes in Math. 227, Springer, Berlin, 1971.

36. M. A. Morrison and J. Brillhart, A method of factoring and the factorization of F_7, Math. Comp., 29 (1975) 183–205.

37. D. A. Plaisted, Fast verification, testing and generation of large primes, Theoret. Comput. Sci., 9 (1979) 1–16; errata, ibid., 14 (1981) 345.

38. S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over GF(p) and its cryptographic significance, IEEE Trans. Inform. Theory, 24 (1978) 106–110.

39. J. M. Pollard, Theorems on factorization and primality testing, Proc. Camb. Philos. Soc., 76 (1974) 521–528.

40. ______, A Monte Carlo method for factorization, BIT, 15 (1975) 331–334.

41. C. Pomerance, Recent developments in primality testing, Math. Intelligencer, 3 (1981) 97–105.

42. ______, On the distribution of pseudoprimes, Math. Comp., 37 (1981) 587–593.

43. ______, Analysis and comparison of some integer factoring algorithms, in Computational Methods in Number Theory (H. W. Lenstra, Jr., and R. Tijdeman, Editors), Math. Centrum Tract 154 (part I), Amsterdam, 1982, pp. 89–139.

44. C. Pomerance, J. L. Selfridge and S. S. Wagstaff, Jr., The pseudoprimes to $25 \cdot 10^9$, Math. Comp., 35 (1980) 1003–1026.

45. C. Pomerance and S. S. Wagstaff, Jr., Implementation of the continued fraction integer factoring algorithm, in Proc. 12th Winnipeg Conf. on Numerical Methods and Computing (1982), to appear.

46. V. R. Pratt, Every prime has a succinct certificate, SIAM J. Comput., 4 (1975) 214–220.

47. M. O. Rabin, Probabilistic algorithms, in Algorithms and Complexity, New Directions and Recent Results (J. Traub, Editor), Academic Press, New York, 1976, pp. 21–39.

48. ______, Probabilistic algorithms in finite fields, SIAM J. Comput., 9 (1980) 273–280.

49. ______, Probabilistic algorithm for primality testing, J. Number Theory, 12 (1980) 128–138.

50. R. Rivest, A. Shamir and L. M. Adleman, A method for obtaining digital signatures and public key cryptosystems, Comm. ACM, 21 (1978) 120–128.

51. C. P. Schnorr, Refined analysis and improvements on some factoring algorithms, J. Algorithms, 3 (1982) 101–127.

52. ______, A Monte Carlo factoring algorithm with finite storage [based on joint work with H. W. Lenstra, Jr.], Seminar on Number Theory, 1981–1982, Exposé #40, Université Bordeaux I, Telence.

53. R. J. Schoof, Quadratic fields and factorization, in Computational Methods in Number Theory (H. W. Lenstra, Jr., and R. Tijdeman, Editors), Math. Centrum Tract 155 (part II), Amsterdam, 1982, pp. 235–286.

54. A. Shamir, Factoring numbers in $O(\log n)$ arithmetic steps, Inform. Proc. Letters, 8 (1979) 28–31.

55. D. Shanks, Class number, a theory of factorization, and genera, in Proc. Symp. Pure Math., vol. 20, Amer. Math. Soc., 1971, pp. 415–440.

56. ______, Five number theoretic algorithms, in Proc. 2nd Manitoba Conf. on Numerical Math. (1972).

57. ______, The infrastructure of a real quadratic field and applications, in Proc. 1972 Number Theory Conf., Boulder, Colorado, pp. 217–224.

58. ______, Square-form factorization, a simple $O(N^{1/4})$ algorithm, to appear.

59. R. Solovay and V. Strassen, A fast Monte-Carlo test for primality, SIAM J. Comput., 6 (1977) 84–85; erratum, ibid., 7 (1978) 118.

60. I. M. Vinogradov, Elements of Number Theory (rev. 5th ed.), Dover, New York, 1954.

61. H. C. Williams, Primality testing on a computer, Ars Combin., 5 (1978) 127–185.

62. ______, A $p + 1$ method of factoring, Math. Comp., 39 (1982) 225–234.

63. M. C. Wunderlich, A running time analysis of Brillhart's continued fraction factoring method, in Number Theory, Carbondale 1979, Lecture Notes in Math. 751, Springer, Berlin, 1979, pp. 328–342.

128. MISCELLANEA

Proper integrals are just like sums, we always say. Are they? Could the equation

$$\int_0^1 x^{-x}dx = \sum_{n=1}^{\infty} n^{-n}$$

be true? See p. 357.