The recipient A knows the deciphering key $(n_A, d_A) = (46927, 26767)$, and so computes 21166^{26767} *mod* $46927 = 16346 =$"YES." How did user A generate her keys? First, she multiplied the primes $p_A = 281$ and $q_A = 167$ to get n_A; then she chose e_A at random (but subject to the condition that $g.c.d.(e_A, 280) = g.c.d.(e_A, 166) = 1$). Then she found $d_A = e_A^{-1}$ *mod* $280 \cdot 166$. The numbers p_A, q_A, d_A remain secret.

In Example 1, how cumbersome are the computations? The most time-consuming step is modular exponentiation, e.g., 16346^{39423} *mod* 46927. But this can be done by the repeated squaring method (see § I.3) in $O(k^3)$ bit operations, where k is the number of bits in our integers. Actually, if we were working with much larger integers, potentially the most time-consuming step would be for each user A to find two very large primes p_A and q_A. In order to quickly choose suitable very large primes, one must use an efficient primality test. Such tests will be described in the next chapter.

Remarks. 1. In choosing p and q, user A should take care to see that certain conditions hold. The most important are: that the two primes not be too close together (for example, one should be a few decimal digits longer than the other); and that $p - 1$ and $q - 1$ have a fairly small g.c.d. and both have at least one large prime factor. Some of the reasons for these conditions are indicated in the exercises below. Of course, if someone discovers a factorization method that works quickly under certain other conditions on p and q, then future users of RSA would have to take care to avoid those conditions as well.

2. In § I.3 we saw that, when n is a product of two primes p and q, knowledge of $\varphi(n)$ is equivalent to knowledge of the factorization. Let's suppose now that we manage to break an RSA system by determining a positive integer d such that $a^{ed} \equiv a$ *mod* n for all a prime to n. This is equivalent to $ed - 1$ being a multiple of the least common multiple of $p - 1$ and $q - 1$. Knowing this integer $m = ed - 1$ is weaker than actually knowing $\varphi(n)$. But we now give a procedure that with a high probability is nevertheless able to use the integer m to factor n.

So suppose we know n — which is a product of two unknown primes — and also an integer m such that $a^m \equiv 1$ *mod* n for all a prime to n. Notice that any such m must be even (as we see by taking $a = -1$). We first check whether $m/2$ has the same property, in which case we can replace m by $m/2$. If $a^{m/2}$ is *not* $\equiv 1$ *mod* n for all a prime to n, then we must have $a^{m/2} \not\equiv 1$ *mod* n for at least 50% of the a's in $(\mathbf{Z}/n\mathbf{Z})^*$ (this statement is proved in exactly the same way as part (a) of Exercise 21 in § II.2). Thus, if we test several dozen randomly chosen a's and find that in all cases $a^{m/2} \equiv 1$ *mod* n, then with very high probability we have this congruence for all a prime to n, and so may replace m by $m/2$. We keep on doing this until we no longer have the congruence when we take half of the exponent. There are now two possibilities:

(i) $m/2$ is a multiple of one of the two numbers $p - 1$, $q - 1$ (say, $p - 1$) but not both. In this case $a^{m/2}$ is always $\equiv 1$ *mod* p but exactly 50%