This set of notes is a companion to chapter 4 in the book. As usual, I will skip many things and mention other things not in the book. In this chapter, the book makes extensive use of abstract algebra. I will try NOT to do this in my approach.

We say that an integer \( x \) has order \( h \) modulo \( m \) if \( h > 0 \), \( a^h \equiv 1 \pmod{m} \), but \( x^k \not\equiv 1 \pmod{m} \) for any integer \( k \) with \( 0 < k < h \). For example, \( 2 \) has order 3 modulo 7 because \( 2^3 \equiv 1 \pmod{7} \) but \( 2^1 \equiv 2 \), \( 2^2 \equiv 4 \), and these are not 1 modulo 7. On the other hand, the order of 3 is 6 modulo 7 because \( 3^1 = 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5, 3^6 = 1 \pmod{7} \). We have some notation for the order of a number. We say \( \text{ord}_m(a) = h \). In our examples above, \( \text{ord}_7(2) = 3 \) and \( \text{ord}_7(3) = 6 \)

**Theorem 1** If \( \text{ord}_m(a) = h \), and \( a^n \equiv 1 \pmod{m} \), then \( h \mid n \).

**Proof:** Let \( n = hq + r \), where \( 0 \leq r < h \). Then \( a^n = a^{hq}a^r \equiv (1)^qa^r \equiv a^r \pmod{m} \). Since \( a^n \equiv 1 \pmod{m} \), we have \( a^r \equiv 1 \pmod{m} \). Since \( r < h \), and \( h \) is the smallest positive integer with \( a^h \equiv 1 \pmod{m} \), \( r \) can not be a positive integer. Since it also can’t be negative, we must have \( r = 0 \), so \( n = qh \), meaning that \( n \) is divisible by \( h \).

Not all numbers have orders. For example, \( 2 \) does not have an order modulo 10 since \( 2^h \) is always even, so it can never be 1 modulo 10. The only numbers that can have orders modulo \( m \) are those numbers relatively prime to \( m \). Moreover, by Euler’s Theorem, if \( \text{gcd}(a, m) = 1 \), then \( a^{\phi(m)} \equiv 1 \pmod{m} \), it follows that all numbers relatively prime to \( m \) do have orders modulo \( m \), and that those orders are divisors of \( \phi(n) \). In our examples above, with \( m = 7 \), \( \phi(7) = 6 \), so all orders for 7 are divisors of 6. In fact, all these divisors are orders:

\[
\text{ord}_7(1) = 1, \quad \text{ord}_7(2) = 2, \quad \text{ord}_7(3) = \text{ord}_7(4) = 3, \quad \text{ord}_7(5) = \text{ord}_7(6) = 6.
\]

If \( \text{ord}_m(a) = \phi(m) \), we say that \( a \) is a **primitive root** modulo \( m \). Thus, 3 and 5 are primitive roots modulo 7. Not all numbers have primitive roots. for example, \( \text{ord}_8(1) = 1 \), \( \text{ord}_8(3) = \text{ord}_8(5) = \text{ord}_8(7) = 2 \). Thus, there is no number with order 4 = \( \phi(8) \). That is, 8 does not have a primitive root. Gauss was the first to answer the question of which numbers have primitive roots. In fact, he proved the following.

**Theorem 2 (Gauss)** The numbers \( n \) that have primitive roots are exactly the numbers 2, 4, \( p^m \) and \( 2p^m \), where \( p \) is an odd prime.

These, and only these numbers have primitive roots. Thus, \( 50 = 2 \cdot 5^2 \) has a primitive root, but \( 35 = 5 \times 7 \) does not. I will not prove Gauss’s theorem, but I will prove a special case: that every prime number has a primitive root. First, we need a few more facts.

**Lemma 1** If \( \text{ord}_m(a) = h \), then \( \text{ord}_m(a^n) = \frac{h}{\text{gcd}(h, n)} \).

**Proof:** Let \( d = \text{gcd}(h, n) \), and let \( n = dn' \), \( h = dh' \). Then \( (a^n)^{h'} = a^{dn'h'} = (a^h)^{n'} \). Since \( a^h \equiv 1 \pmod{m} \), it follows that \( (a^n)^{h'} \equiv 1 \pmod{m} \). This tells us that the order of \( a^n \) must
be a divisor of \( h' \), we want the order to be \( h' \). So let \( k \) be a positive integer and suppose that \((a^n)^k = a^{kn} \equiv 1 \pmod{m}\). Then by Theorem 1, \( h \mid kn \), or \( dh' \mid kdn' \). This means that \( h' \mid kn' \), and since \( \gcd(h',n') = 1 \), it follows that \( h' \mid k \). This shows us that \( h' \) is the smallest positive integer with \((a^n)^{h'} \equiv 1 \) so \( h' \) is the order.

The next result is of importance beyond this section.

**Theorem 3** Suppose that \( f(x) \) is a polynomial with integer coefficients of degree \( n \) and \( p \) is a prime which does not divide the coefficient of \( x^n \) in \( f(x) \). Then \( f(x) \equiv 0 \pmod{p} \) has at most \( n \) solutions for \( x \) with \( 0 \leq x \leq p-1 \). That is, \( f(x) \) can have at most \( n \) solutions modulo \( p \).

**Proof:** We use induction on the degree of \( f(x) \). If \( f(x) \) has degree 1, then \( f(x) = ax + b \), for some integers \( a \) and \( b \), and \( p \nmid a \). Then we can use results from Diophantine equations to show that \( f(x) \) has one and only one zero modulo \( p \). In general, suppose that the result is true for polynomials of degree \( n - 1 \). To step up the degree, suppose that \( f(x) \) has degree \( n \). If \( f(x) \equiv 0 \pmod{p} \) has no solutions, then we are fine \((0 \leq n)\). Suppose, on the other hand, that \( f(a) \equiv 0 \pmod{p} \) for some integer \( a \). Then dividing \( f(x) \) by \( x - a \), we have \( f(x) = (x - a)g(x) + r \), where \( g(x) \) has degree \( n - 1 \) and \( r \) is an integer. Since \( f(a) = r \), we know that \( r \equiv 0 \pmod{p} \), so \( f(x) \equiv (x - a)g(x) \pmod{p} \). Consequently, if \( f(b) \equiv 0 \pmod{p} \), then we would have \((b - a)g(b) \equiv 0 \pmod{p} \), and assuming \( b \neq a \pmod{p} \), then \( g(b) \equiv 0 \pmod{p} \). Note that we are using the fact that \( p \) is a prime here: If \( p \mid (b - a)g(b) \) then \( p \mid (b - a) \) or \( p \mid g(b) \). What this means is that modulo \( p \), the zeros of \( f(x) \) will be \( a \), and any zeros of \( g(x) \). By inductive hypothesis, \( g(x) \) has at most \( n - 1 \) zeros, so \( f(x) \) has at most \( 1 + (n - 1) = n \) zeros.

Next, a cute arithmetic fact.

**Lemma 2** For any positive integer, \( n \),

\[
\sum_{d \mid n} \phi(d) = n.
\]

What this says is that if you add of \( \phi \) applied to all the divisors of \( n \), you get \( n \) back again. For example, if \( n = 20 \), then the possible values of \( d \) are \( 1, 2, 4, 5, 10, 20 \). If we apply \( \phi \) to each of these, we get

\[
\sum_{d \mid 20} \phi(d) = \phi(1) + \phi(2) + \phi(4) + \phi(5) + \phi(10) + \phi(20) = 1 + 1 + 2 + 4 + 4 + 8 = 20.
\]

**Proof:** I like this proof, which uses a very slick idea. Consider the set of fractions

\[
\left\{ \frac{1}{n}, \frac{2}{n}, \frac{3}{n}, \ldots, \frac{n - 1}{n}, \frac{n}{n} \right\}.
\]
There are exactly \( n \) things in this set. Now put each fraction in lowest terms. For example, when \( n = 20 \),

\[
\left\{ \frac{1}{20}, \frac{2}{20}, \frac{3}{20}, \frac{4}{20}, \frac{5}{20}, \frac{6}{20}, \frac{7}{20}, \frac{8}{20}, \frac{9}{20}, \frac{10}{20}, \frac{11}{20}, \frac{12}{20}, \frac{13}{20}, \frac{14}{20}, \frac{15}{20}, \frac{16}{20}, \frac{17}{20}, \frac{18}{20}, \frac{19}{20}, \frac{20}{20} \right\} \\
\rightarrow \left\{ \frac{1}{10}, \frac{2}{10}, \frac{3}{10}, \frac{4}{10}, \frac{5}{10}, \frac{11}{10} \right\} \cup \left\{ \frac{1}{5}, \frac{2}{5}, \frac{4}{5} \right\} \cup \left\{ \frac{1}{2}, \frac{1}{1} \right\} 
\]

which we can write as a union:

\[
\left\{ \frac{1}{20}, \frac{3}{20}, \frac{7}{20}, \frac{9}{20}, \frac{11}{20}, \frac{13}{20}, \frac{17}{20}, \frac{19}{20} \right\} \cup \left\{ \frac{1}{10}, \frac{3}{10}, \frac{7}{10} \right\} \cup \left\{ \frac{1}{5}, \frac{2}{5}, \frac{4}{5} \right\} \cup \left\{ \frac{1}{2}, \frac{1}{1} \right\} 
\]

In general, the denominators are the divisors of \( n \), and for each divisor, \( d \) of \( n \), the numerators for \( d \) are exactly those numbers relatively prime to \( d \). That is, there is a set for each divisor \( d \), and the size of this set will be \( \phi(d) \). Thus, the sum of the sizes of the sets, \( n \), is the sum of \( \phi(d) \) over all divisors \( d \) of \( n \).

We now have the tools to prove our theorem. We will prove slightly more than the existence of primitive roots for primes, we will get a count of how many there are.

**Theorem 4** If \( p \) is a prime, then for each divisor \( d \) of \( p - 1 \), there are \( \phi(d) \) numbers between 1 and \( p - 1 \) with order \( d \) modulo \( p \). In particular, there are \( \phi(p - 1) \) primitive roots for \( p \).

**Proof:** Let \( g(d) \) be the number of numbers \( a \) with \( \text{ord}_p(a) = d \) and \( 1 \leq a \leq p - 1 \). Since \( a^{p-1} \equiv 1 \pmod{p} \) for every such \( a \), every order has to be divisors of \( p - 1 \). This means that \( g(d) = 0 \) unless \( d \mid p - 1 \). Next, consider the congruence \( x^d \equiv 1 \pmod{p} \), or equivalently, \( x^d - 1 \equiv 0 \pmod{p} \). If \( a \) is an element of order \( d \), then \( a^d \equiv 1 \pmod{p} \), so if \( f(x) = x^d - 1 \), then \( a \) is a zero of \( f(x) \). Since \( f(x) \) has degree \( d \), this polynomial can have at most \( d \) solutions. But we can list \( d \) solutions: \( a, a^2, a^3, \ldots, a^d \). Are all these solutions different? Suppose that \( a^i \equiv a^j \pmod{p} \). If \( i > j \) then \( a^{i-j} \equiv 1 \pmod{p} \), but \( \text{ord}_p(a) = d \) means that \( a^k \not\equiv 1 \pmod{p} \) if \( 1 \leq k < d \). This shows that the numbers \( a^i \) above are all distinct modulo \( p \), so we have an entire list of solutions for \( f(x) \).

From Lemma 1, \( \text{ord}_p(a^k) = \frac{d}{\gcd(d,k)} \), so those \( a^k \) with \( k \) relatively prime to \( d \) will all have order \( d \). This means that there are exactly \( \phi(d) \) elements of order \( d \), provided there is an element of order \( d \). What we have shown so far: \( g(d) = 0 \) if \( d \nmid p - 1 \) If \( d \mid p - 1 \) then it is still possible that \( g(d) = 0 \), but if \( g(d) \neq 0 \), then \( g(d) = \phi(d) \). Since every number between 1 and \( p - 1 \) has an order, we also know that the sum of all \( g(d) \)'s is \( p - 1 \). That is, we know \( \sum_{d \mid p - 1} g(d) = p - 1 \). Since \( \sum_{d \mid p - 1} \phi(d) = p - 1 \) and \( g(d) \leq \phi(d) \) for all \( d \), we must have \( g(d) = \phi(d) \) for all \( d \). That is, it can not be the case that \( g(d) = 0 \) for some divisor of \( p - 1 \), or else the sum of the \( g \)'s would be less than the sum of the \( \phi \)'s. Consequently, there are exactly \( \phi(d) \) numbers of order \( d \) for each divisor \( d \) of \( p - 1 \), and in particular, there are
exactly $\phi(p-1)$ primitive roots.

It is worth mentioning that a good piece of Gauss’s theorem now follows. By the Chinese Remainder Theorem, if $m$ and $n$ are relatively prime, then $a^h \equiv 1 \pmod{mn}$ is equivalent to the system \[
\begin{align*}
\{ a^h &\equiv 1 \pmod{m} \\
 a^h &\equiv 1 \pmod{n} \}
\end{align*}
\] Also, if $\text{ord}_m(a) = h_1$ and $\text{ord}_n(a) = h_2$, then $\text{ord}_{mn}(a) = \text{lcm}(h_1, h_2)$ (I will let you prove this last fact). Because of this, for every $a$, $\text{ord}_n(a) < \phi(n)$ whenever $n$ is divisible by two distinct odd primes, or when $n$ is divisible by 4. The reason for this is that if $p$ and $q$ are odd primes, then $p-1$ and $q-1$ are both even, as is $\phi(4)$. For example, suppose that $a$ is a primitive root for both 7 and 11. This means that $\text{ord}_7(a) = 6$ and $\text{ord}_{11}(a) = 10$. Consequently, $\text{ord}_{77}(a) = \text{lcm}(6, 10) = 30$, but $\phi(77) = 60$. This means that $n$ can’t have a primitive root unless $n$ is divisible by at most one odd prime number, or if $4 \mid n$ and $n > 4$. The only possibilities that remain are $n = 2, 4, p^m$, or $2p^m$. Also, if $a$ is a primitive root for $p^m$ and $a$ is odd, then $a$ is also a primitive root for $2p^m$. This can be used to show that $2p^m$ has primitive roots whenever $p^m$ does. To prove Gauss’s theorem, all that is missing is to show that $p^m$ has primitive roots for all positive $m$ and all odd primes $p$. I believe our book has a proof, but I will not finish it here.

Primitive roots give us a new tool to use. Here are two applications of primitive roots.

Primality Proving

Our first application of primitive roots will be to give a method for actually proving a number is prime. Till now, all we could do to prove a number, $n$, is prime is to use trial division by primes up to $\sqrt{n}$, which is hopelessly slow for numbers of any size. Otherwise, we could show that a number is “probably” prime, by showing that $a^{n-1} \equiv 1 \pmod{n}$. We can prove that $n$ is prime if we can find a number $a$ for which $\text{ord}_n(a) = n-1$, since $\phi(n) = n-1$ only if $n$ is prime. So, we can prove that $n$ is prime if we can find a primitive root for $n$.

For example, consider $n = 97$. We seek a number $a$ for which $\text{ord}_{97}(a) = 96$. Given any $a$, the first step is probably to check $a^{96} \pmod{97}$, because if we don’t get 1, we can stop, the number can’t be prime in that case. In fact, $2^{96} \equiv 1 \pmod{97}$, of course. Next, we know that the order of 2 must be a divisor of 96, so we need to rule out possible smaller orders: 1, 2, 3, 4, 6, 8, 12, 16, 24, 32, 48. This is a long list, but there is a trick: If $\text{ord}_{97}(a)$ does not divide either 32 or 48, then it must be 96 (the only divisor of 96 which does not divide 32 or 48 is 96 itself). To check if $\text{ord}_{97}(a)$ divides 32 or 48, we simply calculate $a^{32} \pmod{97}$ and $a^{48} \pmod{97}$. If neither is 1, then the order does not divide the exponent. Unfortunately, $2^{48} \equiv 1 \pmod{97}$, so the order of 2 DOES divide 48. In fact, it turns out that its order is 48. So we failed! This is common, and all it means is that 2 is not a primitive root. Next, try $a = 3$. Again, we find that $3^{48} \equiv 1 \pmod{97}$, so we move to the next value. We can skip $a = 4$. If $a$ is not a primitive root, then no power of $a$ can be a primitive root. Moving to $a = 5$, $5^{32} \equiv 35 \pmod{97}$ and $5^{48} \equiv 96 \pmod{97}$. Thus, $\text{ord}_{97}(5) = 96$, so 5 is a primitive root, and 97 is prime. In general,

**Theorem 5** Let $p$ be an odd prime and suppose that $p \nmid a$. If for every prime divisor $q$ of $p-1$, $a^{(p-1)/q} \not\equiv 1 \pmod{p}$, then $a$ is a primitive root for $p$. 

Page 4
Proof: If a is not a primitive root, then the order of a is a proper divisor of $p - 1$. Thus, if a has order $d$, then $dk = p - 1$ for some integer $k > 1$. Let $q$ be a prime divisor of $k$, so $k = qm$ for some integer $m$. Now $p - 1 = dk = dqm$, so $d$ is a divisor of $\frac{p - 1}{q}$. In particular, this means that $a^{(p-1)/q} \equiv 1 \pmod{p}$. What we have shown: If the order of $a$ is not $p - 1$ then there is a prime $q \mid p - 1$ for which $a^{(p-1)/q} \equiv 1 \pmod{p}$. Consequently, if $a^{(p-1)/q} \not\equiv 1 \pmod{p}$ for all such $q$, then the order of $a$ can't be less than $p - 1$.

Corollary 1 If $n$ is a positive integer and $a$ has the property that $a^{n-1} \equiv 1 \pmod{n}$ but for each prime divisor $q$ of $n - 1$, $a^{(n-1)/q} \not\equiv 1 \pmod{n}$, then $n$ is prime.

Proof: As in our example of 97, if $a^{n-1} \equiv 1 \pmod{n}$ then the order of $a$ must be a divisor of $n - 1$. If $a^{(n-1)/q} \not\equiv 1 \pmod{n}$ for each prime divisor $q$ of $n - 1$ then the order of $a$ must be $n - 1$, so $\phi(n) = n - 1$, proving that $n$ is prime.

How long does it take to find a primitive root? If 2 is not a primitive root, then neither is $2^2$ or $2^3$, etc. It is possible for 6 to be a primitive root even if neither 2 nor 3 is, however. If you believe the Extended Riemann Hypothesis, then every prime $p$ has a primitive root $a$ satisfying $a < 2(\ln(p))^2$.

There is another issue, however, in proving that a number $p$ is prime: our method of proving that $a$ is a primitive root of $p$ requires us to produce a factorization of $p - 1$. If $p$ is a very large number, factoring $p - 1$ can be very hard. For example, consider the number $n = 2^{329} + 39$. Maple reports that $n$ is a probable prime. Suppose we would like to prove that it is prime. By the above, we would need to factor $n - 1 = 2^{328} + 38$. Using Maple’s ifactor routine, we get an answer like:

$$\text{ifactor}(n - 1, \text{easy})$$

$$(2)(5)^2(7)(11)_c92(5101),$$

meaning that Maple found several small prime divisors of $n - 1$ but that this still left a 92-digit number, which Maple determined to be composite. Factoring a general 92-digit number can take a while.

Indices and the discrete logarithm problem

Primitive roots have applications to cryptography. Suppose that $p$ is a prime, with primitive root $g$. Then every number between 1 and $p - 1$ is a power of $g$. For example, 5 is a primitive root for 23, and the powers of 5 (mod 23) are:

<table>
<thead>
<tr>
<th>$x$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5^x$</td>
<td>5</td>
<td>2</td>
<td>10</td>
<td>4</td>
<td>20</td>
<td>8</td>
<td>17</td>
<td>16</td>
<td>11</td>
<td>9</td>
<td>22</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x$</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5^x$</td>
<td>18</td>
<td>21</td>
<td>13</td>
<td>19</td>
<td>3</td>
<td>15</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>14</td>
<td>1</td>
</tr>
</tbody>
</table>
From this table, we see that $5^9 \equiv 11 \pmod{23}$, and so on. Also, if someone asks for the solution to $5^x \equiv 15 \pmod{23}$, then we could scan the table to find $x = 17$, meaning $x \equiv 17 \pmod{22}$, since $5^{17+22k} \equiv 5^{17}(5^{22})^k \equiv 15(1)^k \equiv 15 \pmod{23}$.

In general, if $p$ is a prime and $g$ is a primitive root for $p$, the question of finding $x$ so that $g^x \equiv a \pmod{p}$ is called the **discrete logarithm problem**. The reason is that we are asking about an exponent, very much like if we were asked to find $x$ so that $5^x = 17$, where the answer would be $x = \log_5 17$. In general, if $p$ is large, solving $g^x \equiv a \pmod{p}$ for $x$ is thought to be a hard problem, just as factoring a large number is thought to be hard. It should be pointed out again that just because people think a problem is hard does not make it so. Still, you could become famous by finding an easy way to solve the general problem of $g^x \equiv a \pmod{p}$ for $x$.

When ever a problem is considered hard, people try to use it to generate cryptosystems. To see why, we introduce some cryptographic terminology. A function $f(x)$ from a set $S$ to a set $T$ is called a **one-way** function if

1) it is one-to-one,
2) $f(x)$ is “easy” to calculate,
3) for most values of $a$, it is “hard” to solve $f(x) = a$ for $x$.

Related to a one-way function is a trapdoor function. A trapdoor function is a function $f(x)$ for which it is hard to solve $f(x) = a$ without extra knowledge about $f$.

An example of a trapdoor function is the one used in the RSA cryptosystem: given a large number $n$, known not to be prime, and an integer $e$ relatively prime to $\phi(pq)$, $f(x) \equiv x^e \pmod{n}$. In this case, if you have the additional knowledge that $n = pq$ for two primes $p$ and $q$ (that is, if you know how $n$ factors), then you can solve the congruence $ed \equiv 1 \pmod{\phi(n)}$, and the solution to $f(x) = a$ is simply $a^d \pmod{n}$. Trapdoor functions are required for public key cryptosystems where you want the entire world to know how to send you a coded message, but you want to be the only one able to decode the message.

The difference between a trapdoor function and a one-way function is that there is no (known) easy way to solve $f(x) = a$ for $x$ with a one-way function. That is, being “in the know” will not help speed up the calculation. One-way functions have a very common use in computer systems: when a system administrator stores passwords for the users on a network, it would be nice if you didn’t have to worry about someone hacking into the password file and uncovering all the passwords. One way to do this is to not store the passwords, but to store instead, a one-way function applied to the passwords. That is, convert a password, $w$, to a number, and then store $f(w)$. When someone logs on and gives their password, then what you do is compare $f($password$)$ to $f(w)$. If it matches, you figure the person knows the password. Since the file contains $f(w)$ instead of $w$, the only way for someone to get the password from $f(w)$ is to solve the problem $f(x) = a$, where $a$ is what is recorded in the password file. Both UNIX and VMS operating systems use this idea.
Whereas the function $f(x) \equiv x^e \pmod{n}$ is a trapdoor function, the function $f(x) \equiv g^x \pmod{p}$ is thought to be an honest one-way function. Here are some cryptosystems that have been built upon the discrete logarithm problem.

**The Diffie-Hellman key selection protocol.**

How might two people decide on a private key over an insecure channel? Here is a method proposed by W. Diffie and W. E. Hellman (1976). First, you and your party agree on a prime $p$ and a primitive root $g$. The prime should be large, certainly greater than $10^{40}$. Next, you privately pick a random number, $x$, and send your party the number $g^x$. Your party picks a random number, $y$, and sends you $g^y$. Now you construct the private key $g^{xy} = (g^y)^x$, and your party constructs the same private key $g^{xy} = (g^x)^y$. All these computations are done modulo $p$. What the outside world sees is $g, p, g^x$ and $g^y$. Since they don’t know either $x$ or $y$, presumably, they cannot construct $g^{xy}$, so they do not know your private key.

As a very simple example, suppose the prime is 23 with primitive root 5. If you pick $x = 3$, then you send $5^3 \equiv 10 \pmod{23}$. Say your friend selects 9 and sends you $5^9 \equiv 11 \pmod{23}$. Then you construct the key by taking your friend’s number, 11 and cubing it: $11^3 \equiv 20 \pmod{23}$, and your friend constructs the key by taking the number you sent, 10, and raising it to the 9’th power: $10^9 \equiv 20 \pmod{23}$. That is, you both agree on the key 20.

Here is a more realistic example: You wish to send a message to a friend, say “Do you know the answer to question 3?” This translates into:

46881098889410848788961093817810748792967891109388109094789293828887102941,

a 74-digit message. To send this message to your friend, you both agree on a large prime, say

$$p = 10^{74} + 207$$

(possibly a poor choice) and a primitive root $g = 3$. I am not sure 3 is a primitive root, but I think it is. In practice, this is probably good enough. Next, you pick a random number, say 12345, and calculate $3^{12345} \pmod{p}$ to get

$$A = 1863359963619159544262521729497524098276582212838872313532582876893904337$$

You send this to your friend. Your friend picks a random number, say 11223, and calculates $3^{11223} \pmod{p}$ or

$$B = 73878641578748055418751379768859616272738272992567530735975619073453351.$$  

The key will be $K = 3^{12345 \times 11223}$, which you calculate via $K = A^{12345}$, and which your friend calculates via $K = B^{11223}$. The key is

$$K = 4607196454614546309820971568775118507708928756269337520440317091356703073.$$
Once you have agreed on a key, send the message $M + K \pmod{p}$, or
\[
51488295344025395098782065386585867300676819865657446615229610920243806014
\]
Your friend translates by subtracting $K \pmod{p}$.

The ElGamal cryptosystem

Again, we fix a large prime $p$ and a primitive root $g$. Pick a random number $a$ with $0 < a < p - 1$, and calculate $K \equiv g^a \pmod{p}$. To send a message, $M$ to someone, chose a random integer $k$ and send the pair $(g^k, Mg^ak)$. If your friend knows $a$, then they can recover the message as follows: $g^ak \equiv (g^k)^a \pmod{p}$, and once they have calculated $g^ak$, they calculate its multiplicative inverse, $x \pmod{p}$, and get $M$ via the formula $M \equiv Mg^ak \times x \pmod{p}$. This system requires that both you and your friend know $p$, $g$, and $a$. The rest of the world might know $g$ and $p$, but they can’t know $a$ or they can read the message as well.

The Massey-Omura cryptosystem

As before, we start by having two people agree on a large prime $p$ with primitive root $g$. Next, pick a random number $a$ with $\gcd(p - 1, a) = 1$, and solve $ab \equiv 1 \pmod{p - 1}$ for $b$. Your friend picks a random number $c$ relatively prime to $p - 1$ and solves $cd \equiv 1 \pmod{p - 1}$ for $d$. Given a message $m$, send your friend $M^a \pmod{p}$. Your friend does not know $a$ and cannot translate this message. However, they take the message and raise it to the $c$ power, and send you $M^{ac} \equiv (M^a)^c \pmod{p}$. You take this new message and raise it to the $b$ power, sending $M^{abc} \equiv (M^{ac})^b \pmod{p}$ back to your friend. Now $ab \equiv 1 \pmod{p - 1}$ so $ab = 1 + k(p - 1)$ for some integer $k$. This means $M^{abc} = M^{c+k(p-1)} = M^c(M^k)^{p-1} \equiv M^c \pmod{p} \equiv M^c \pmod{p}$, by Fermat’s Little Theorem. So your friend now has $M^c$, and can recover the message because by a similar computation, $M^{dc} \equiv M \pmod{p}$.

I have heard this idea described analogously as follows: suppose you want to get something secret to a friend across a crowded room. Take it, put it in a box, and put a lock on the box (raise $M$ to the $a$ power). Pass this from person to person till it gets to your friend. Your friend does not have a key to the lock. They put a second lock of theirs on the box (raise to the $c$ power) and pass the box back to you. Now you can safely take your lock off (raise to the $b$ power) because their lock is still in place, and when the box gets back to them, they remove their lock to get at the contents.

Big Step/ Little Step

One method for solving $g^x \equiv a \pmod{p}$ for $x$ is to just try $x = 0, 1, 2, \ldots$ until they get $a$. This is, of course, quite slow, and takes, on average $\frac{1}{2}p$ steps. Here is an alternative. Let $m = \lfloor \sqrt{p} - 1 \rfloor$. Then $x = qm + r$, where $0 \leq q \leq m$ and $0 \leq r < m$. Also, $g^x = (g^m)^qg^r$. We can find $x$ if we can find $q$ and $r$. We do this as follows: Compute $g^m$, $g^{2m}$, ..., $g^{m^2}$. This is called the big step. We could calculate $g, g^2, \ldots, g^{m-1}$, and compute all products in the
two lists till we got $a$, but this would be no better than the first method. Instead, calculate $y = g^{-1} \pmod{p}$. That is, find $y$ so that $gy \equiv 1 \pmod{p}$ and calculate $a, ay, ay^2, \ldots$, comparing these to the powers of $g^m$. Once you find a match, say $ay^r = (g^m)^q$, then we have found $q$ and $r$, so we know $x$.

For example, suppose we wish to solve $2^x \equiv 25 \pmod{37}$. Here, $m = \lceil \sqrt{37} \rceil = 6$. We have $2^6 \equiv 27$, $2^{12} \equiv 26$, $2^{18} \equiv 36$, $2^{24} \equiv 10$, $2^{30} \equiv 11$, $2^{36} \equiv 1 \pmod{37}$. In this case, $y \equiv 2^{-1} \equiv 19$. I got this by the cheap trick that $2^{-1} \pmod{p}$ is the reciprocal of $2$ in $\mathbb{Z}_p^*$. In general, this trick only works for 2. However, if $p + 1$ is divisible by 3 then $3^{-1} \equiv \frac{p + 1}{3} \pmod{p}$, and if not, $3^{-1} \equiv \frac{2p + 1}{3} \pmod{p}$, and this idea could be extended, at least if $g$ is small. Now we take 25 and start multiplying by 19 till we get something on the list \{ 27, 26, 36, 10, 11, 1 \}. Since 25 is not on the list, we calculate $25 \times 19 = 31$, and this isn’t on the list either. Continuing, $31 \times 19 \equiv 34, 34 \times 19 \equiv 17, 17 \times 19 \equiv 27$, and 27 is on the list. This means that $25 \times 2^{-4} \equiv (2^6)^4 \equiv 2^{24} \equiv 10 \pmod{37}$, so $x = 9$. In general, this method takes no more than $2m$ steps to find $x$, with $\frac{3}{2}m$ being the average number of steps needed. There are other, more elaborate schemes for solving for $x$, but this is the simplest.

To give some idea of how hard the discrete logarithm problem is, here are some extra credit problems, worth 3 points each. Each has the same flavor: I give you a prime $p$, a primitive root $g$, and an number $a$. You are to solve $g^x \equiv a \pmod{p}$ for $x$.

1. $p = 83$, $g = 2$, $a = 27$. For this, you have to use a calculator and the big step/little step method.

2. $p = 10^{10} + 19$, $g = 2$, $a = 111$. Again, this is too small to do by anything other than the big step/little step method. Can you write some code in Mathematica/Alpha to do this?

3. $p = 3 \times 10^{30} + 91$, $g = 3$, $a = 488723603054929248287646942294$. I’ve given this problem for more than 15 years without anyone solving it! With today’s computers, it should not be too bad, however.

4. $p = 4 \times 10^{50} + 57$, $g = 5$, $a = 234875656810706951458302741301382804433197596608712$. I assume this will be out of range for a few more years.