You may use computer algebra systems such as mathematica, gp, or sage or program in C, Java, or Python. Please submit your code if any) as part of your homework. If you do, make sure that your programs compile and run correctly; my students will not debug your programs. The program should also be humanly readable.

1. $3 \in \mathbb{F}_{1013}^*$ generates a group of order $1012 = 4 \cdot 11 \cdot 23$. Solve the discrete logarithm problem $g = 3$, $h = 321$ by using the Pohlig-Hellman attack, i.e. find an integer $0 < a < 1012$ such that $h = g^a$ by computing first $a$ modulo $2, 4, 11,$ and $23$ and then computing $a$ using the Chinese Remainder Theorem.

2. Use the schoolbook version of Pollard’s rho method to attack the discrete logarithm problem given by $g = 3$, $h = 245$ in $\mathbb{F}_{1013}^*$, i.e. find an integer $0 < a < 1012$ such that $h = g^a$, using the $t_i$ and $r_i$ (the twice as fast walk) as defined in class (aand repeated here). Let $t_0 = g$, $a_0 = 1$, and $b_0 = 0$ and define

$$t_{i+1} = \begin{cases} t_i \cdot g \\ t_i \cdot h \\ t_i^2 \end{cases}, a_{i+1} = \begin{cases} a_i + 1 \\ a_i \\ 2a_i \end{cases}, b_{i+1} = \begin{cases} b_i \\ b_i + 1 \\ 2b_i \end{cases} \text{ for } t_i \equiv \begin{cases} 0 \mod 3 \\ 1 \mod 3 \\ 2 \mod 3 \end{cases},$$

where one takes $t_i$ as an integer. The twice as fast walk has $r_i = t_{2i}$.

The walk could start at any $t_0 = g^{a_0} h^{b_0}$ for known $a_0$ and $b_0$ – but then the homework would be harder to correct.

3. Use factor base $\mathcal{F} = \{2, 3, 5, 7, 11, 13\}$ to solve the DLP $h = 281$, $g = 2$, in $\mathbb{F}_{1019}^*$. I.e. pick random powers of $g = 2$, check whether they factor into products of powers of $2, 3, 5, 7, 11,$ and $13$; if so, add a relation to a matrix. The columns of the matrix correspond to the discrete logs of $2, 3, 5, 7, 11,$ and $13$. Once you have 6 rows try to solve the matrix; note that these computations take place modulo the group order 1018. It might be that some of the rows are linearly dependent, in that case you need to generate another relation. Once you have all discrete logs of the primes in the factor base, check whether $h$ is smooth and if not find a $h/g^i$ (for some $i$) which is smooth. $2, 3, 5, 7, 11,$ and $13$; if so, add a relation to a matrix. The columns of the matrix correspond to the discrete logs of $2, 3, 5, 7, 11,$ and $13$. Once you have 6 rows try to solve the matrix; note that these computations take place modulo the group order 1018. It might be that some of the rows are linearly dependent, in that case you need to generate another relation. Once you have all discrete logs of the primes in the factor base, check whether $h$ is smooth and if not find a $h/g^i$ (for some $i$) which is smooth.

E.g. $2^{291} \equiv 52 \mod 1019$; over the integers $52 = 2^2 \cdot 13$, so we include the relation $291 \equiv 2a_2 + a_{13} \mod 1018$. Note that you can run into difficulties inverting modulo 1018 since it is not prime. E.g. $2^{658} \equiv 729 \mod 1019$; over the integers $729 = 3^6$, so we include the relation $658 \equiv 6a_2 \mod 1018$ but 6 is not invertible modulo 1018 and we can only determine $a_3 \equiv 449 \mod 509$ and need to test whether $a_3 = 449$ or $a_3 = 449 + 509$. Here $2^{140} \equiv 1016 \mod 1019$ and $2^{449+509} \equiv 3 \mod 1019$, thus $a_3 = 958$.

Hint: if you’re using Pari-GP you’ll find

```
factor(lift(Mod(2^i,p)))
```

a usefull command.
Background information:
The Pohlig-Hellman attack attack works in any group and is a way to reduce the reduce the hardness of the DLP to the hardness of the DLP in subgroups of prime order. In particular you'll see in the exercise that it works against the DLP in $\mathbb{F}_{1013}^*$ by solving DLPs in groups of size 2, 11, and 23. Here is the general description:

Let $G$ be a cyclic group generated by $g$ and let the challenge be to find $\log_g h = a$. Let the group order $n$ factor as $n = \prod_{i=1}^{e_i} p_i^{e_i}$ where $p_i \neq p_j$ for $i \neq j$. Then $a$ can be computed from the information

\[
a \equiv a_1 \mod p_1^{e_1} \\
a \equiv a_2 \mod p_2^{e_2} \\
a \equiv a_3 \mod p_3^{e_3} \\
\vdots \\
a \equiv a_r \mod p_r^{e_r}
\]

by using the Chinese remainder theorem. This is because the $p_i^{e_i}$ are coprime and their product is $n$. So, if one can find the DL modulo all $p_i^{e_i}$ one can compute the entire DL.

Put $n_i = n/p_i^{e_i}$. Since $g$ has order $n$ the element $g_i = g^{n_i}$ has order $p_i^{e_i}$. The element $h_i = h^{n_i}$ is in the subgroup generated by $g_i$ and it holds that $h_i = g_i^{a_i}$, where $a_i \equiv a \mod p_i^{e_i}$.

E.g. $\mathbb{F}_{16}^* = \langle g \rangle$ has 15 elements, so one can first solve the DLP $h = g^a$ modulo 3 and then modulo 5. For such small numbers one can simply compute $h^5$ and compare it to 1, $g^5$, and $g^{10}$ to find whether $a$ is equivalent to 0, 1, or 2 modulo 3. Then one compares $h^3$ to 1, $g^3$, $g^6$, $g^9$, and $g^{12}$ to see whether $a$ is congruent to 0, 1, 2, 3, or 4 modulo 5.

The same approach works also for $\mathbb{F}_{17}^*$ which has 16 = $2^4$ elements – but here one can do much better! Write $a = a_0 + a_1 2 + a_2 2^2 + a_3 2^3$. Then $h^8$ is either equal to 1 or to $-1 = g^8$ depending on whether $a_0$ is 0 or 1. Once that result is known we can compare $(h/g^{a_0})^4$ with 1 and $-1$ to find $a_1$ etc. So we can solve a much smaller DLP. Instead of going for a modulo $p_i^{e_i}$ at once we can first obtain $a$ modulo $p_i$, then modulo $p_i^2$, then modulo $p_i^3$, etc. till $p_i^{e_i}$ by each time solving a DLP in a group of size $p_i$.

In general, for each $p_i$ in the factorization of $n$ one does the following:

1. Put $h' = h$, $a_{i,-1} = 0$
2. for $j = 0$ to $e_i - 1$
   (a) put $h' = h/(g^{a_{i,j-1}p_i^{j-1}})$ //using precomputed $g^{-1}$
   (b) solve the DLP of order $p_i$ for $a_{i,j} = \log_{g^{a_{i,j}p_i^{-1}}} (h'^{n/p_i^{j+1}})$

and then combine the $a_{i,j}$ to $a_i = \sum_{j=0}^{e_i-1} a_{i,j}p_i^j$ and then those $a_i \mod p_i^{e_i}$ (using CRT) to $a \mod n$.

**Important:** the Pohlig-Hellman attack handles one prime at a time, not a prime power. That means that your DL table has only $p_i$ elements and that you solve $e_i$ DLSs in subgroups of order $p_i$. You can see the difference in the example with $\mathbb{F}_{17}^*$ below. xs

Numerical examples:

$\mathbb{F}_{11}^* = \langle 2 \rangle$, find $a$ so that $3 = 2^a$. So $g = 2$ and $h = 3$. Compute $n_1 = 10/2 = 5$, $g^{n_1} = 2^5 = -1$, and $h^{n_1} = 3^5 = 1$ to see that $a \equiv 0 \mod 2$. Then compute $n_2 = 10/5 = 2$, ...
\(g^{n_2} = 2^2 = 4, \quad g^{2n_2} = 2^4 = 5, \quad g^{3n_2} = 2^6 = 9, \quad \text{and} \quad g^{4n_2} = 2^8 = 3\) and compare that to \(h^{n_2} = 3^2 = 9\) to see that \(a \equiv 3 \mod 5\). These two congruences imply that \(k = a\) and indeed \(g^8 = h\).

\[\mathbb{F}_{17}^* = \langle 3 \rangle,\] find \(a\) so that \(7 = 3^a\). So \(g = 3\) and \(h = 7\). In this example we will obtain \(a\) one bit at a time. First compare \(h^8 = 7^8 = -1\) to 1 and \(-1\) to see that \(a \equiv 1 \mod 2\). Then compute \(h/g = 8\) and then \((h/g)^4 = -1\), so also the next bit is 1 and we see \(a \equiv 3 \mod 4\). Then compute \(h/g^3 = 16\) and then \((h/g^3)^2 = 1\) to see that the next bit is 0, so \(a \equiv 3 \mod 8\). Finally, since \(h/g^3 = 16 = -1\) we see that the highest bit is 1, so \(a \equiv 11 \mod 16\) and indeed \(3^{11} = 7\). This solved the DLP in \(\mathbb{F}_{17}^*\) with just 4 very easy computations and comparisons. So computing DLs in fields \(\mathbb{F}_p\) with \(p = 2^r + 1\) is easy.