The Baby-Step-Giant-Step algorithm.

The Baby-Step-Giant-Step algorithm is an algorithm introduced by Dan Shanks in 1969, which can be applied to solve the discrete logarithm problem in a cyclic group.

Let G be a cyclic group with n elements, and let $a \in G$ be a generator of the group. It means that $G = \{a,\ a^2,\ \dots,\ a^n = e\}$. In particular, every $x \in G$ can be written as $x = a^s$, for some $s \in \mathbf{Z}$. The exponent s, which by Lagrange's theorem it is only well defined modulo n, is *by definition* the discrete logarithm of x in base a

$$s := \log_a(x) \bmod n.$$

The Baby-Step-Giant-Step algorithm is a deterministic algorithm for computing the discrete logarithm in an arbitrary finite cyclic group. It exploits the fact that every element $x \in G$ can be written as

$$x = a^{j+mi}, \tag{1}$$

for integers $m,\ i,\ j$ satisfying $m \sim \sqrt{n}$, and $0 \le i,\ j \le m$. Equation (1) can be rewritten as $a^i = xa^{-mj}$. Then the logarithm $\log_a(x)$ is obtained by comparing two lists: the baby steps a^i and the giant steps xa^{-mj}, for $0 \le i, j \le m$. When a coincidence is found between the two lists, namely one has $a^{i_0} = xa^{-mj_0}$ for some i_0 and j_0, then

$$\log(x)_a = i_0 + mj_0.$$

By BSGS, one obtains the desired logarithm by computing at most $2m \sim 2\sqrt{p}$ powers modulo p and comparing the two lists. By the naif method one could possibly have to compute up to p powers modulo p, before obtaining the desired logarithm.

Example. Fix $p = 433$ and let $a = 7$ be a primitive root in $\mathbf{Z}_p^*$. We want to calculate the discrete logarithm of $x = 166$ in base a. In this case, $m = 21 \sim \sqrt{433}$.
We first produce the list of the **Baby-Steps**:

$$a^i \bmod p, \qquad \text{for } 0 \le i \le m-1$$

$a^0 = 1$
$a^1 = 7$
$a^2 = 49$
$a^3 = 343$
$a^4 = 236$
$a^5 = 353$
$a^6 = 306$
$a^7 = 410$
$a^8 = 272$
$a^9 = 172$
$a^{10} = 338$
$a^{11} = 201$
$a^{12} = 108$
$a^{13} = 323$
$a^{14} = 96$
$a^{15} = 239$
$a^{16} = 374$
$a^{17} = 20$
$a^{18} = 140$
$a^{19} = 114$
$a^{20} = 365$

$a^{-m} = a^{-21} = 292$

Next we produce the list of the **Giant-Steps**:

$$xa^{-mj} \bmod p, \quad \text{for } 0 \le j \le m-1$$

and each time we check whether the value the new Giant-Step already appears in the list of the Baby-Steps. When that is the case, we are done.

$x \cdot a^0 = 166$
$x \cdot a^{-21} = 409$
$x \cdot a^{-42} = 353$!!!

We have found a coincidence between the two lists: $a^5 = x \cdot a^{-42}$. This means that

$$x = a^{5+42} = a^{47} \quad \text{and} \quad \log_7(166) = 47.$$

Indeed one can check that $7^{47} = 166 \bmod 433$.

The Pohlig-Hellman algorithm.

Let G be a cyclic group of order N and suppose that $N = \prod_i q_i^{e_i}$ is the product of small distinct primes q_i, for $i = 1, \ldots, s$. Then $G \cong G_1 \times \ldots \times G_s$, with

$$\# G_i = q_i^{e_i} \quad \text{and} \quad G_i \cong Z_{q_i^{e_i}}.$$

By the Chinese Remainder Theorem the discrete logarithm problem in G can be reduced to the discrete problem in the smaller groups G_i. Hence *the essential case is* $G = \mathbf{Z}_{q^e}$*, for* q *odd prime and* $e \ge 1$.

Let P be a generator of G and let Q be a given element. Then $Q = kP$, for some integer $k \bmod q^e$. We want to determine k, which by definition is the discrete logarithm of Q in base P. Recall that the subgroups of G are linearly ordered

$$0 = q^e G \subset q^{e-1} G \subset \ldots \subset qG \subset G,$$

where $q^m G$ is the q^{e-m}-torsion subgroup, for $m = 0, 1, \ldots, e$.

The Pohlig-Hellman algorithm provides a method to solve the DLP in G. Write k in base q, as $k = k_0 + k_1 q + \ldots + k_s q^s$, for $k_j \in \{0, \ldots, q^e - 1\}$. Then

$$Q = kP = k_0 P + k_1 qP + \ldots + k_s q^s P, \qquad (*)$$

where the summand $k_m q^m P$ is an element in the q^{e-m}-torsion subgroup of G, for $m = 0, 1, \ldots, e$.

In order to determine the coefficients k_m, we precompute the elements of the q-torsion

$$T = \{0, q^{e-1}P, \ldots, (q-1)q^{e-1}P\}.$$

By multiplying both terms of the equation $(*)$ by q^{e-1} we get

$$q^{e-1}Q = k_0 q^{e-1} P,$$

which is an element in the q-torsion T. By comparing it with the elements of T, we determine k_0. In general, once we have determined $k_0, \ldots, k_{j-1}$, we obtain k_j as follows: we multiply both terms of the equation

$$Q - k_0 P - \ldots - k_{j-1} q^{j-1} P = k_j q^j P + \ldots + k_s q^s P$$

by q^{e-j-1}. The only surviving element on the right hand side is $k_j q^{e-1} P$. By comparing it with the elements of T, we determine k_j.