Long Jump Home

First, let's understand the criterion for being able to reach point L from point 0 using a set of jump lengths $x_1, x_2, \ldots, x_n$. From the extended Euclidean algorithm, it is not difficult to understand that from point 0, we can reach point $g = \gcd(x_1, x_2, \ldots, x_n)$ using a linear combination of jumps. Obviously, after that, we can visit all coordinates of the form $g \cdot k$, where $k \in \mathbb{Z}$.

Next, we will show that the points described above are all the points that can be visited with such a set of x values. Indeed, each of $x_1, x_2, \ldots, x_n$ is divisible by g, thus the coordinate after a jump will always be a multiple of g.

After that, we want point L to be equal to $g \cdot k$ for some integer k. Thus, the necessary and sufficient condition for the set of x values is: $L \vdots \gcd(x_1, x_2, \ldots, x_n)$.

Then the problem reduces to finding a set of x values with the minimum cost such that its gcd divides L. Therefore, from the current set, we are only interested in its current gcd and cost. Let's set up a dynamic programming approach:

$dp_{i,z}$ — the minimum cost of a set formed only from the first i abilities that has gcd $= z$. We will understand how to recalculate such a DP: if the current ability has parameters x_i, c_i, then the recalculation from the previous layer will be as follows:

$$dp_{i,\gcd(w,x_i)} = \min(dp_{i,\gcd(w,x_i)}, dp_{i-1,w} + c_i), \qquad 0 \le w \le L$$

Thus, the DP works in $\mathcal{O}(n \cdot L \cdot \log C)$. How to find the answer? We need to look at all DP values of the form $dp_{n,d}$, where d divides $|L|$.