**Example 2.** Let the $v_i$ be as in Example 1, and take $V = 24$. Then, working from right to left in our 5-tuple $\{2, 3, 7, 15, 31\}$, we see that $\epsilon_4 = 0$, $\epsilon_3 = 1$ (at which point we replace 24 by $24 - 15 = 9$), $\epsilon_2 = 1$ (at which point we replace 9 by $9 - 7 = 2$), $\epsilon_1 = 0$, $\epsilon_0 = 1$. Thus, $n = (01101)_2 = 13$.

We now describe how to construct the knapsack cryptosystem (also called the Merkle–Hellman system). We first suppose that our plaintext message units have $k$-bit integers $P$ as their numerical equivalents. For example, if we're working with single letters in the 26-letter alphabet, then every letter corresponds to one of the 5-bit integers from $0 = (00000)_2$ to $25 = (11001)_2$ in the usual way.

Next, each user chooses a superincreasing $k$-tuple $\{v_0, \ldots, v_{k-1}\}$, an integer $m$ which is greater than $\sum_{i=0}^{k-1} v_i$, and an integer $a$ prime to $m$, $0 < a < m$. This is done by some random process. For example, we could choose an arbitrary sequence of $k + 1$ positive integers $z_i$, $i = 0, 1, \ldots, k$, less than some convenient bound; set $v_0 = z_0$, $v_i = z_i + v_{i-1} + v_{i-2} + \cdots + v_0$ for $i = 1, \ldots, k - 1$; and set $m$ equal to $z_k + \sum_{i=0}^{k-1} v_i$. Then one can choose a random positive $a_0 < m$ and take $a$ to be the first integer $\geq a_0$ that is prime to $m$. After that, one computes $b = a^{-1}$ *mod* $m$ (i.e., $b$ is the least positive integer such that $ab \equiv 1$ *mod* $m$), and also computes the $k$-tuple $\{w_i\}$ defined by $w_i = av_i$ *mod* $m$ (i.e., $w_i$ is the least positive residue of $av_i$ modulo $m$). The user keeps the numbers $v_i$, $m$, $a$, and $b$ all secret, but publishes the $k$-tuple of $w_i$. That is, the enciphering key is $K_E = \{w_0, \ldots, w_{k-1}\}$. The deciphering key is $K_D = (b, m)$ (which, along with the enciphering key, enables one to determine $\{v_0, \ldots, v_{k-1}\}$).

Someone who wants to send a plaintext $k$-bit message $P = (\epsilon_{k-1}\epsilon_{k-2} \cdots \epsilon_1\epsilon_0)_2$ to a user with enciphering key $\{w_i\}$ computes $C = f(P) = \sum_{i=0}^{k-1} \epsilon_i w_i$, and transmits that integer.

To read the message, the user first finds the least positive residue $V$ of $bC$ modulo $m$. Since $bC \equiv \sum \epsilon_i b w_i \equiv \sum \epsilon_i v_i$ *mod* $m$ (because $bw_i \equiv bav_i \equiv v_i$ *mod* $m$), it follows that $V = \sum \epsilon_i v_i$. (Here we are using the fact that both $V < m$ and $\sum \epsilon_i v_i \leq \sum v_i < m$ to convert the congruence modulo $m$ to equality.) It is then possible to use the above algorithm for superincreasing knapsack problems to find the unique solution $(\epsilon_{k-1} \cdots \epsilon_0)_2 = P$ of the problem of finding a subset of the $\{v_i\}$ which sums exactly to $V$. In this way we recover the message $P$.

Note that an eavesdropper who knows only $\{w_i\}$ is faced with the knapsack problem $C = \sum \epsilon_i w_i$, which is *not* a superincreasing problem, because the superincreasing property of the $k$-tuple of $v_i$ is destroyed when $v_i$ is replaced by the least positive residue of $av_i$ modulo $m$. Thus, the above algorithm cannot be used, and, at first glance, the unauthorized person seems to be faced with a much more difficult problem. We shall return to this point later.

**Example 3.** Suppose that our plaintext message units are single letters with 5-bit numerical equivalents from $(00000)_2$ to $(11001)_2$, as above. Suppose that our secret deciphering key is the superincreasing 5-tuple