$$\sum_{i \in [n]} min[S_i, max[D - s(A), 0]]x_i \geq D - s(A)$$

1. When $s(A) \geq D$, the right hand side of the inequality is either zero or negative. If $x_i \geq 0$, then the left hand side is non-negative and thus always greater than or equal to the right hand side.

2. For the dynamic programming, we declare a two dimensional array, a, as shown below. Each cell represents the maximum $\sum_{i \in A} min[s_i, D - k]x_i$, given the set A to be pumps 1 through i and the sum $s(A) = k$. n denotes the total number of pumps. Some constraint is violated only when the value is maximized.

$i \mid s(A)$	k_1	k_2	k_3	...	k
0					
...					
$n-1$					
n	-∞	-∞	-∞	-∞	0

   The base cases are $a[n][j] = -\infty$ for all $j < k$, and $a[n][k] = 0$.

   Then, for each pump, the DP algorithm either picks or not pick it. For each column of $s(A) = j$, we start from the bottom of the column and iterate based on the expression

$$a[i][j] = max[a[i+1][j + s_i] + s_i x_i, a[i+1][j]],$$

   which has considered all the possible DP choices. After we decide on pump i, we are left with a subproblem without external constraints. If we can solve the subproblem optimally, because addition of gallons is linear, we obtain an optimal solution to the larger problem.

   At the end, in order to record the set A, we iterate over a certain column and capture the jumps in values. Because k is bounded by n, the running time of the DP algorithm is polynomial in terms of n.