This problem can be regarded as a SLS problem. Since the last line is not included for the optimization, we simply let $w_1, \cdots, w_n$ be the reverse order of the words (i.e., the first word is w_n and the last word is w_1). For the first line, it starts with word w_n and end with word w_j. So the sub-problem can be built on $w_1, w_2, \cdots, w_{j-1}$ for recursion.

Pre-compute: Denote the slack of a line which contains $w_i, \cdots w_j$ as S_{ij} $(i \leq j)$. If the words exceed the limit of L, S_{ij} is assigned to ∞. S_{ij} should be computed before recursion process.

According to the problem description and lecture, the optimal solution will be:

$$OPT(n) = \min_{1 \leq i \leq n} S_{i,n}^2 + OPT(i-1) \tag{1}$$

The algorithm is stated as follow:

Algorithm 1 Algorithm for problem 1

Pre-compute all S_{ij}
Set $OPT(0) = 0$
For $j = 1, 2, \cdots n$
  Use the recurrence to compute $OPT(j) = \min_{1 \leq i \leq j} S_{ij}^2 + OPT(i-1)$
Endfor
Return $OPT(n)$

Proof of Correctness:
The correctness of this algorithm has been proved by defining the sub-problems, declaring the recurrence and base cases which is shown above the algorithm. Also, the correctness of algorithm for SLS problems have been proved during the lecture.

Time complexity analysis:
For the pre-computing, for each i, all S_{ij} can be computed in $O(n)$ assuming j is in increasing order, so all S_{ij} can be computed in $O(n^2)$.

For the recurrence, each iteration can be computed in $O(n)$. Since there are n iterations in total, the total running time is $O(n^2)$.