Paragraph Breaking

A paragraph is a sequence of words, each of which is a string of symbols. A modern text formation program will print the paragraph so that it looks "nice," in particular, that the right margin is even. How is that done? The answer is by using an appropriate paragraph breaking algorithm, which is a dynamic program.

Let $w_1, w_2, \ldots w_n$ be the words of the paragraph, and let ℓ_i be the length of w_i.

Here is a simple algorithm. Suppose each line is restricted to 80 symbols. The algorithm simply chooses each line to be have the maximum length that is less than 80. For example, if $\ell_1 + \cdots + \ell_i <= 80$ and $\ell_1 + \cdots + \ell_{i+1} > 80$, the last word on the first line will have be w_i, and the next line begins at w_{i+1}, and so forth. This algorithm is greedy, and takes $O(n)$ time. But a modern paragraph breaking algorithm is more sophisticated.

Let us suppose that the ideal line length has length 80, but that the editor can squeeze or stretch a line that is longer or shorter, but that the line is not a nice looking. We will say that a line is "ideal" if it has length 80, and that otherwise, we assign the line a "penalty" which is related to how far the line deviates from ideal. To make the problem simple, we will assign a quadratic penalty, namely $(L - 80)^2$, to a line of length L, except that the last line can have length shorter than 80 with no penalty.

The problem is to design a dynamic program which breaks the paragraph into lines in such a way as to minimize the sum of the penalties of the lines.

The output of the algorithm is a sequence of numbers, such as (21, 18, 22, 20, 6), which indicates that the first line has 21 words, the second line 18 words, and so forth; note that the last line is shorter.

Time Complexity

The greedy algorithm takes only $O(n)$ time, but does not find the optimal breaking. You can write a fairly simple dynamic program that finds the optimal solution in $O(n^2)$ time. There are faster, more sophisticated algorithms, however.

I do not want you to write compilable computer code for your algorithm. What you hand in must precisely describe the algorithm, however; perhaps in pseudo-code, perhaps simply in English, or perhaps a mixture of the two.