

Instructions

1. You are encouraged to work in groups of two. If you cannot find a partner, you can work alone.
2. Please submit one copy of the assignment; if you are working with a partner, both names should appear on the assignment.
3. For **Part A** of the assignment, you must submit an electronic copy of your Java application via WebCT (by the time of the lecture on the due date of the assignment).

Part A

A text consists of a sequence of words, $W = \{w_1, w_2, \dots, w_n\}$. We may assume the words are over the ASCII alphabet. Let c_i be the number of characters of the word w_i , i.e., $c_i = |w_i|$. We also have a maximum line length L .

We want to write a pretty-printer for text. A formatting of W consists of a partition of the words in W into lines. In the words assigned to a single line, there should be a space after each word except the last; and so if w_j, w_{j+1}, \dots, w_k are assigned to one line, then we should have:

$$\left[\sum_{i=j}^{k-1} (c_i + 1) \right] + c_k \leq L.$$

We will call an assignment of words to a line valid if it satisfies this inequality. We call the slack of a line the number of spaces left at the right margin.

Write and implement an efficient algorithm to find a partition of a set of words W into valid lines, so that the sum of the squares of the slacks of all lines (including the last line) is minimized.

The input should be a text file, where words are assumed to be separated by one or more spaces. The output should be pretty-printed to the standard output. For example, on the input given on the left, we get the output given on the right.

```
Call me Ishmael.
Some years ago,
never mind how long precisely,
having little or no money in my purse,
and nothing particular to interest me on shore,
I thought I would sail about a little
and see the watery part of the world.
```

```
Call me Ishmael. Some years ago, never
mind how long precisely, having little
or no money in my purse, and nothing
particular to interest me on shore, I
thought I would sail about a little
and see the watery part of the world.
```

(For 1% of the final grade, where is this text from?)

Part B

- (a) Suppose you are given a connected graph G , with edge costs that are all distinct. Prove that G has a unique minimum cost spanning tree.
 - (b) A graph G may have many minimum cost spanning trees when the edge costs are not all distinct. Given any MCST T , show how to compute a new cost function c' to the edges so that the costs of all the edges are distinct and the output of Kruskal's algorithm on G with the new costs is T .
- Suppose you have n video streams that need to be sent, one after another, over a communication link. Stream i consists of a total of b_i bits that need to be sent, at a constant rate, over a period of t_i seconds. You cannot send two streams at the same time, so you need to determine a schedule for the streams: an order in which to send them. Whichever order you choose, there cannot be any delays between the end of one stream and the start of the next. Suppose your schedule starts at time 0 (and hence ends at time $\sum_{i=1}^n t_i$, whichever order you choose). We assume that all the values of b_i and t_i are positive integers.

There is a fixed parameter r which imposes the following constraint:

() For each natural number $t > 0$, the total number of bits you send over time interval from 0 to t cannot exceed rt .*

Note that this constraint is only imposed for time intervals that start at 0, not for time intervals that start at any other value.

We say that a schedule is valid if it satisfies the constraint (*) imposed by the link.

Give an algorithm that takes a set of n streams, each specified by its number of bits b_i and its time duration t_i , as well as a link parameter r , and determines whether there exists a valid schedule. The running time of your algorithm should be polynomial in n .