Algorithms in the Real World (15-853), Fall 08
Assignment #3 Due: Oct 23

You can look up material on the web and books, but you cannot look up solutions to the given problems. You can work in groups, but must write up the answers individually.

Problem 1: Solving a Recurrence (5pt)
For the space-efficient Edit Distance problem (Lecture 3), we used the recurrence:

$$\begin{aligned} T(n, m) &= T(n/2, k) + T(n/2, m-k) + O(mn) \\ T(1, m) &= m \\ T(n, 1) &= n \end{aligned}$$

Give a formal proof that $T(m, n) = O(mn)$.

Problem 2: Interleaving Subsequences (10pt)
Given two strings S_1 and S_2 and a text T, you want to find whether there is an occurrence of S_1 and S_2 *interwoven* in T, possibly with spaces. E.g., the strings $S_1 =$ `banana` and $S_2 =$ `tantan` occur interwoven in $T =$`tabaranantangna` as follows:

```
  **  **     **
tabaranantangna
++      ++++
```

(Note that each letter has either a `*`, a `+`, or neither, but not both.) Give an efficient algorithm for this problem (i.e. one that is polynomial in the size of the inputs).

Problem 3: "Truncated Edit Costs" (10pt)
Consider the following edit-distance cost model: (a) each insertion or deletion costs one unit; (b) however, if there are more than k consecutive insertions, or k consecutive deletions, they cost only k units. Give an algorithm that finds the minimum edit distance under this cost model in time $O(nm)$. Note that the time should not depend on k. (Do not worry about space efficiency for this problem).

Problem 4: Minimum Cover (10pt)
Given strings A and B, a minimum cover of A by B is a decomposition $A = w_1 w_2 \cdots w_k$ where each w_i is a substring of B and k is minimum. Show how to compute a minimum cover (given that one exists) in $O(|A| + |B|)$ time.