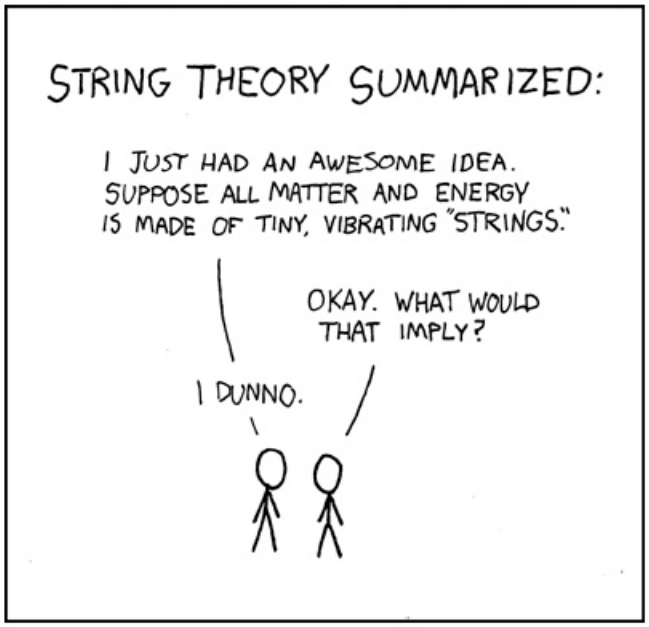
# CS302 - Assignment 20

Due: Tuesday, May 8 at the beginning of class
Hand-in method: paper




http://xkcd.com/171/

1. **[6 points]** In class, we mentioned including the "swap" operation in our edit distance function. The swap operation only swaps *adjacent* characters AND any given character may only be swapped once. For example, EDIT('recieve','receive') = 1 if we allow the swap operation (swapping 'i' and 'e'). However, we could not use two swaps to transform 'recieve' into 'recevie' because this would involve swapping 'i' twice, which is not allowed.

   (a) Write the recursive subproblem relation for swap (e.g. for insert, it was $\text{EDIT}(X, Y) = 1 + \text{EDIT}(X_{1\ldots n}, Y_{1\ldots m-1})$ and for delete it was $\text{EDIT}(X, Y) = 1 + \text{EDIT}(X_{1\ldots n-1}, Y_{1\ldots m})$).

   (b) Describe how to change the EDIT procedure covered in class to include the swap operation. You may either edit the pseudocode directly or describe what needs to be added/changed.

2. **[11 points]** String matching algorithms

   (a) **[6 points]** In class, we discussed three different string matching algorithms: naive, FSA based and Rabin-Karp. For each of these three algorithms, list two things, 1) describe a situation where the algorithm would perform better than the other two algorithms 2) describe an application where this situation would occur.

   (b) **[5 points]** Many other string matching algorithms also exists besides those mentioned in class. Investigate a new string matching algorithm not discussed in the class or the book. Give a short summary (in your own words) of how the algorithm works and, as in part a) of this problem, state in what situation the new algorithm would perform better and an application where this would occur. Be sure to cite your sources!

3. **[8 points]** You are asked to write a new string class/data structure that must support the following operations and average case runtime restrictions:

   - length - $O(1)$
   - concatenate - $O(n + m)$
   - substitute - $O(j)$ where $j$ is the number of characters changed in the string

   Describe a data structure that supports these string operations and describe how you would implement each of these operations. Note: you will need to keep around additional data besides the string itself. Be sure you are clear about how you are storing the string and this additional data.

4. **[5 points]** Someone gives you a large list of single "terms" that have been entered into a search engine of size $n$. You would like to try and identify terms that are possible spelling mistakes of each other. You decide to use edit distance and would like to find all pairs of words in the list that are within edit distance 2 of eachother (by the traditional definition of edit distance with inserts, deletes and substitutions). The naive algorithm would take each word and calculate the edit distance with the remaining $n - 1$ words, but for large $n$ this can be very slow.

   Can you come up with a more efficient approach? Your algorithm can make compromises and sometimes miss terms that should have been found, but you should try and avoid this.

   Note there is no single one right answer to this problem since this is a hard problem, however, you should give some justification for why your approach is better than the naive approach.