58093 String Processing Algorithms (Autumn 2013)
Exercises 1 (October 29)

Solve the following problems before the exercise session and be prepared to present your solutions at the session.

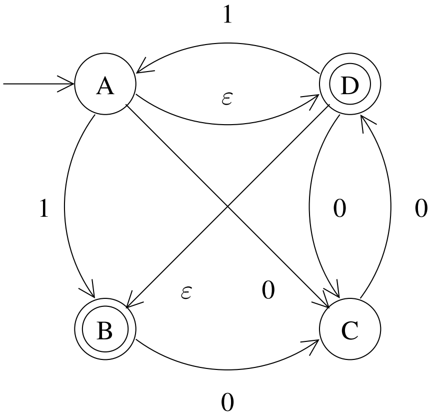
1. The number of comparison operations required to sort n items is $\Omega(n \log n)$. Recall the proof of this fact and be prepared to present the proof.

2. The worst case time complexity of the standard quicksort algorithm is $\Omega(n^2)$, but by a suitable pivot selection one can achieve $\mathcal{O}(n \log n)$ time. Explain how to achieve

   (a) average time complexity $\mathcal{O}(n \log n)$
   (b) expected time complexity $\mathcal{O}(n \log n)$
   (c) worst case time complexity $\mathcal{O}(n \log n)$.

3. A full binary tree is a binary tree where every node is either a leaf or has two children. Show that every full binary tree with n leaves has exactly $2n - 1$ nodes. *Hint: Use induction.*

4. Transform the following nondeterministic finite automaton into a deterministic finite automaton.



5. Write a program that computes the ten most frequent words in the english text file
   `http://pizzachili.dcc.uchile.cl/texts/nlang/english.50MB.gz`
   Note that the file is compressed with `gzip` and must be decompressed before processing.
   You may consider the space character to be the only word separator, i.e., words can contain punctuation, newlines etc., but using a more sophisticated word parsing method is allowed too.
   Be prepared to show the code and the output of your program.

6. The Fibonacci numbers are defined using the recurrence

$$
\begin{aligned}
f_0 &= 0 \\
f_1 &= 1 \\
f_n &= f_{n-1} + f_{n-2}
\end{aligned}
$$

   Write two programs that compute f_n for a given n using

   (a) a recursive function based directly on the recurrence
   (b) dynamic programming.

   What is the largest n for which you can compute f_n using program (a) in less than five seconds?