Exercise 13: Rot13 is an encryption method in which each of the 26 characters a-z and A-Z is replaced by the character 13 places ahead or 13 places back in the alphabet. Write a program that reads a text and outputs the rot13 equivalent. Example:

Enter a text, line by line. Finish with an empty line.
> This text should be impossible to read
> because it has been encrypted with a fine
c> encryption method.
> Guvf grkg fubhyq or vzcbffvoyr gb ernq
or phfr vg unf orra rapelcgqrq jvgu n svar
rapelcgvba zrgubq.

Note that since rot13 is symmetric, the same program can be used both for encoding and decoding.


Exercise 14: Make regular expressions for the following string descriptions:
1. Strings containing only a's and b's.
2. Strings that do not contain white space.
3. Strings with exactly one word regardless of white space.
4. Like 1. but the number of a's should be even.
5. Any string.
6. No string, not even the empty string.

Exercise 15: Design regular expressions for positive integers, arbitrary integers and decimal numbers.
Extra Credit: Start with the design of a regexp for arbitrary floats, including exponents. The standard solution involves concepts beyond the current coverage of the course.

Exercise 16: Use a regular expression to parse the string "Date: 2001-11-09" such that the variables $year, $month and $day are assigned to correctly.

Exercise 17: The restriction enzyme AaaI cuts DNA after every C that is followed by "GGCCG". The shorthand notation for this is Cˆ GGCCG". Write a program that reads DNA and identifies all cuts.
Extra Credit: Generalize your program to accommodate more restriction enzymes, and ultimately, to read restriction enzyme data from an appropriate input file at http://rebase.neb.com/rebase/rebase.html.

G. Fuellen (responsible),
J. Clausen