1. Write regular expressions for the following languages whose alphabet is $\Sigma = \{0, 1\}$:

(a) All strings.
(b) No strings.
(c) The empty string.
(d) The string 011.
(e) The strings 0 and 011.
(f) All strings beginning with a 1.
(g) All strings beginning with a 1 and ending with a 0.
(h) All strings that containing exactly three 1’s.
(i) All strings in which every 0 is immediately preceded by and followed by a 1.

2. Design deterministic finite automata (DFA) to recognise the following languages whose alphabet is $\Sigma = \{0, 1\}$:

(a) Every occurrence of the substring 11 is followed by a 0.
(b) Every third symbol is a 1.
(c) All strings of 0’s and 1’s with an even number of 0’s and an even number of 1’s ($\epsilon$ included).

Note: When building a DFA (or NFA) by hand, it is a good idea to annotate each state with a statement that describes the kind of strings that can possibly reach that state.

3. Construct non-deterministic finite automata (NFA) from the following regular expressions:

(a) $(a|b)^*$
(b) $a^*|b^*$
(c) $a|(b)c^*$
(d) $a(a|b)^*b$

Note: You may use Thompson’s subset construction algorithm.

4. Regular expressions in the real world:

(a) Use egrep to print all words starting with a y and ending with a y from /user/dicts/words.
(b) Write a sed script to reverse the content of each line.
(c) Write a perl script to replace all occurrences of a string in a file with another string.