TDDD14/TDDD85 Formal Languages and Automata Theory
Assignment 1 – 2017
Deadline Wed. 2017-04-19, 17:00

For all the problems below it is not sufficient just to give a solution. You must **justify your answers**. In the final exam unexplained answers will be granted 0 points. You may write your solutions in english or swedish.
Hand in your solutions to
- Christer Bäckström: for TDDD85 (U1)
- Jonas Wallgren: for TDDD14 (others)
Please mark clearly which course you are taking. You have three options:
- Hand in your solutions at a lecture or tutorial.
- Put your solutions in the box "Post till IDA" in front of the Café Java in the B building.
- Email a single pdf file (no other formats, please) to the appropriate teacher and start the subject line with the course code.

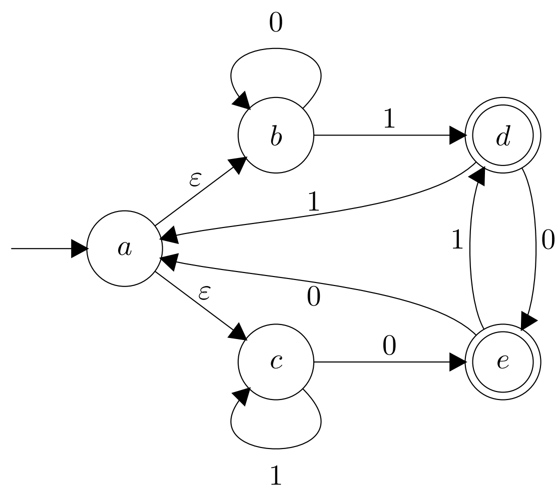
1. For each of the five regular expressions to the left, tell which ones of the five strings to the right it accepts (note that a regular expression may match more than one string).

1) $0^*(0+1)(0+1)^*$	a) 0011
2) $(0+1)^*1(0+1)^*0(0+1)^*$	b) 1001
3) $(0+(0+1)^*)(0^*+1^*)$	c) 010101
4) $(0^*+1)(0+1^*)$	d) 00001111
5) $(0^*+1^*)^*$	e) ε

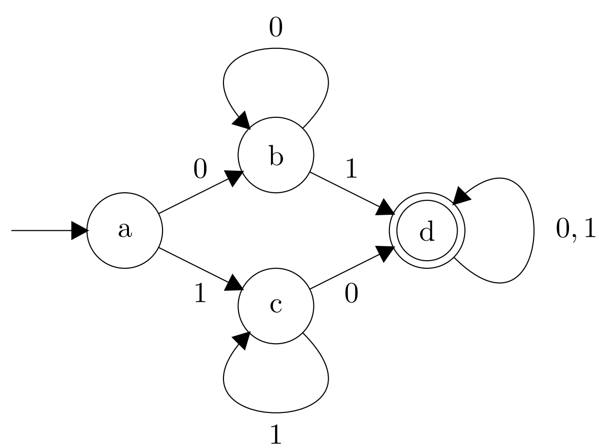
2. Consider the usual syntax for floating-point numbers.
   - A number may start with an optional $+$ or $-$ sign.
   - No decimal point or decimals is necessary if the number is an integer, but if there is a decimal point, then it must be followed by at least one decimal.
   - The integer part can be omitted if it is zero
   The following are examples of floating point numbers:
   12.34 -0.57 .236 +1.23 -.44

   Let the alphabet consist of the digits $0, 1, \ldots, 9$, the decimal point "."and the signs "+" and "−". Construct a DFA that accepts all floating-point numbers and nothing else. You may use the label "$0-9$" on transitions as a shorthand for "$0, 1, \ldots, 9$".

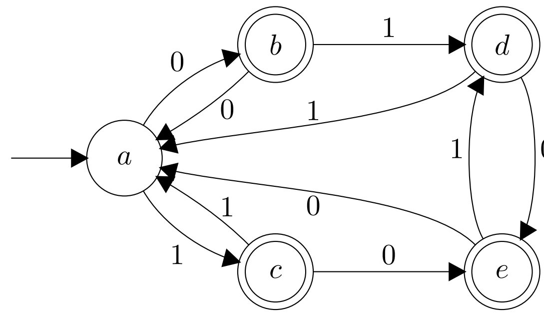
3. Convert the following NFA to an equivalent DFA, using the subset construction method.



4. Convert the following DFA to an equivalent regular expression by using the GNFA construction method or some of the other standard methods in the course.



5. Use the minimization algorithm to construct a DFA that is equivalent to the following DFA and that has a minimal number of states.



6. Consider the following two languages over the alphabet $\Sigma = \{a, b\}$:

$$L_1 = \{w \in \Sigma^* \mid \#a(w) > \#b(w) \text{ and } \#b(w) \leq 3\}$$
$$L_2 = \{w \in \Sigma^* \mid \#a(w) > \#b(w) \text{ or } \#b(w) \leq 3\}$$

(Recall that $\#a(w)$ denotes the number of occurences of symbol a in w, and analogously for $\#b(w)$.)

(a) Show that L_1 is regular.

(b) Show that L_2 is not regular. (Note that a string can have more than 3 occurences of b if it satisfies the first condition).

For each of L_1 and L_2, show that it is regular or show that it is not regular.

7. The *Thue-Morse sequence* is an infinite sequence $w_1, w_2, w_3, \ldots$ of strings over the alphabet $\{0, 1\}$, recursively defined as follows:

$$w_1 = 0$$
$$w_{i+1} = w_i\overline{w_i} \text{ for all } i \geq 1$$

where $\overline{w}$ denotes the bitwise complementary string, i.e. 0's are replaced by 1's and vice versa. For example, the first four strings are:

$$w_1 = 0$$
$$w_2 = w_1\overline{w_1} = 0\overline{0} = 01$$
$$w_3 = w_2\overline{w_2} = 01\overline{01} = 0110$$
$$w_4 = w_3\overline{w_3} = 0110\overline{0110} = 01101001$$

Define the language $T = \{w_1, w_2, w_3, \ldots\}$, i.e. T consists of all strings in the Thue-Morse sequence.

(a) Use the pumping lemma to show that T is not a regular language.

(b) Use the third claim in the Myhill-Nerode theorem to show that T is not a regular language, i.e. show that the equivalence relation

$$x \equiv_T y \text{ iff for all } \; z \in \Sigma^* (xz \in T \Leftrightarrow yz \in T)$$

has an infinite number of equivalence classes.

(Hint for both a and b: Note that all strings in T satisfy that $|w_{i+1}| = 2|w_i|$, i.e. $|w_i| = 2^{i-1}$).