CS 205 (Borgida)   Fall 2016
Homework 8

DUE: MONDAY MORNING 5:00am NOVEMBER 28 (note the one day delay).
BUT NO LATE ASSIGNMENTS ACCEPTED AFTER TUESDAY 5am.

PART A on Finite Automata

Please read the lecture notes on Finite Automata, and Hamburger & Richards Sakai>Resources>_10 (Also maybe the examples in Rosen 13.3, although I do not emphasize the construction of deterministic automata from start, since we have a construction that takes a non-determninistic automaton without null moves and returns a deterministic one.) Please be sure you understand what it means for a DFA or NFA to "recognize a language (set of strings)".

1. In each of the following questions you are asked to draw a finite automaton recognizing a language over alphabet V={b,c}. (Remember to indicate the start state and final/accepting states.) The number of states and transitions must be as small as possible (we will not be able to grade complicated answers). To help this, please do not add the sink/dead states to your diagram, unless you feel it is absolutely necessary.

i) set of strings over {b,c} sarting with c
ii) set of strings over {b,c} having exactly one c
iii) set of strings over {b,c} that have 2 or more c
iv)* set of strings described by the regular expression b*c*

[For extra credit & practice, give regular expressions for languages i), ii), iii). Again, only the shortest and clearest answers will be graded.]

2. Consider the NFA N3 with transitions over alphabet {0,1} and states A,B,C, with start state A, final state C.

	0	1
A	A,B	A
B	C	
C		C

0) draw the graph of the corresponding automaton
i) Describe briefly in Enlgish the language recognized by N3.
ii) Describe this language using a _simple_ regular expression
iii) Construct the DFA corresponding to NFA N3 using to the technique described in lecture and resource _10 (also sketched in the textbook).

3. Construct an NFA, possibly with null/lambda moves, that recognizes exactly the language described by the regular expression (ab* | ac*)*. You can do so by (i) describing the language in English and then constructing the smallest and clearest automaton that recognizes it, or, (ii) for full credit, by following exactly the general algorithm taught in class (See lecture notes).

In the first case, FAs that are not obviously correct/incorrect will not be graded.

##
Part B on next page

1. In the handout you have seen the function $sqr(x) = x^2$ over the positive integers, and the corresponding relation R_{sqr}. Let's now also define the function $plusThree(x) = x + 3$ over the positive integers.

   (a) Give 2-tuples in the relation $R_{plusThree}$ corresponding to $x = 1, 2, 3, 4$.

   (b) Give at least four 2-tuples in the composition $R_{sqr} \circ R_{plusThree}$

   (c) Give at least four 2-tuples in the composition $R_{plusThree} \circ R_{sqr}$

   (d) Mathematicians use the function rather than relation notation. For them, composing $plusThree()$ and $sqr()$ is the function which maps x to $plusThree(sqr(x))$.[1] Write the first four 2-tuples in the relation $R_{plusThree(sqr())}$ corresponding to the cases when x=1,2,3,4.

   (e) From this example, how does function composition $f \bullet g$, relate to relation composition between R_f and R_g?

2. For each of the following relations over the positive integers write a row indicating if it is (Y) or not (N): reflexive, symmetric, transitive, antisymmetric

   $R1 = \{(a, b) \mid a + b \text{ is even}\}$
   $R2 = \{(a, b) \mid a \times b\text{is even}\}$
   $R3 = \{(v, w) \mid$ v and w are strings over the letters A-Z, and v is the reverse of w$\}$

   You do not have to prove Y answers, but must give a counter-example for each N answer.

3. Prove formally that relation composition is associative. In other words, if R, S and T are relations over A, then $R\circ(S\circ T) = (R\circ S)\circ T$.

   Your proof will have the form

```
(a,b) in  R o (S o T)
iff  there exists y in A such that ...      [by  definition of o]
...
iff   (a,b) in (R o S) o  T                 [by definition of  o]
```

[1] Let's write $plusThree \bullet sqr$ for this kind of mathematical function composition, which you encounter in Section 3.4 of your textbook.