AMERICAN UNIVERSITY OF ARMENIA
College of Science and Engineering
CS 310 Theory of Computing
INC Completion Set

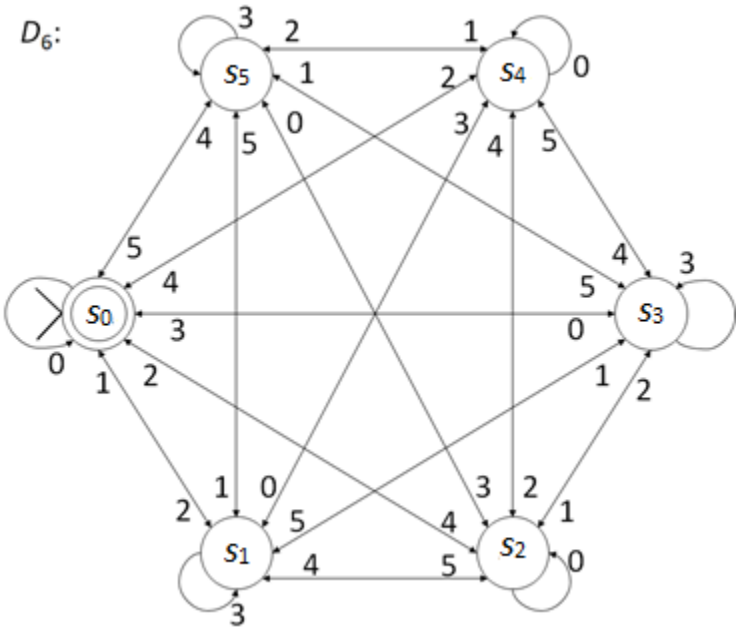
Mail to: skhachat@aua.am

Textbook: E. Kinber, C. Smith. "Theory of Computing: A Gentle Introduction", 2000

Section 1. Finite Automata (chapter 2)

Task 1: Construct in the alphabet $\Sigma_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ a DFA $A_8 = \{Q, \Sigma_{10}, s, \delta, F\}$ that checks if the input can be divided by 8. It accepts integer numbers in decimal notation written in the reversed order. For example, a number 1234 is written on the input tape as a sequence 4, 3, 2, 1 and rejected. While a number 1432 is written as a sequence 2, 3, 4, 1 and accepted. Recall that a number is divided by 8, if its lowest 3 digits as a 3-digit number (the first 3 cells on the input tape) can be divided by 8.

Task 2: Consider below a DFA $D_6 = \{Q, \Sigma_6, s_0, \delta, F\}$ in the alphabet $\Sigma_6 = \{0, 1, 2, 3, 4, 5\}$ that checks if the input can be divided by 6. Apply the Algorithm 2.7.1 (page 51) to minimize its states.



Task 3: Cyclic numbers are natural numbers (possibly very long), the sequence of digits of which exhibits the following property – if multiplied by the lowest digit, called the base digit, the digits in the result form the same sequence except the last one that jumps to the beginning. For example, in the decimal numeral system the cyclic number ending with 4 is 102564. Indeed, $102564 \times 4 = 410256$.

There is a simple algorithm that computes digit-by-digit a cyclic number with the given base digit:

1. Initialize the carry with 0;
2. Multiply the last digits by itself and add the carry to the result;
3. Assign the units of the obtained result to the next digit;
4. Assign the tens of the obtained result to the carry;
5. Multiply the next digits by the last digit and add the carry;
6. Return to step 3;
7. Stop the process when the last two digits are 10 (the carry remains 0).

It is obvious that 0 and 1 are cyclic numbers by themselves.

Construct in the alphabet $\Sigma_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ a DFA $C = \{Q, \Sigma_{10}, s, \delta, F\}$ that checks if the input is a cyclic number. It accepts integer numbers in decimal notation written in the reversed order. For example, the following input words are accepted: “0”, “1”, “111”, “465201”.

Hint: consider an indexed set of states $s_{bcd} \in Q$, where three indexes b, c and d represent values of the base digit, the current carry and the next digit respectively.

Section 2. Pushdown Automata and Context-free Grammars (chapter 3)

Task 4. Consider as a template a pushdown automaton that accepts the language $L = \{a^m b^m \mid m = 0, 1, 2, \ldots\}$: $A(L) = \{\{s, t, f\}, \{a, b\}, \{a\}, \Delta, \{s, f\}\}$, where Δ is defined as

$((s, a, e), (t, a))$
$((t, a, e), (t, a))$
$((t, b, a), (f, e))$
$((f, b, a), (f, e))$

a) Construct a pushdown automaton that accepts the language $L_a = \{a^m b^{xm} \mid m = 0, 1, 2, \ldots\}$, where x is the last digit of your ID# that is greater than 1.
b) Construct a pushdown automaton that accepts the language $L_b = \{a^{xm} b^m \mid m = 0, 1, 2, \ldots\}$ with the same x from item a). Assume only one character may be read at the stack top.
c) Write the context-free grammars for the languages L_a and L_b.

Task 5: Consider the famous problem of Hanoi Towers. There are 3 pegs enumerated 0, 1 and 2, and N discs of increasing sizes. Initially, all discs are placed at a peg i (i = 0, 1, or 2) in the ascending order: the largest disc at the bottom and the smallest one – at the top. The task is to move all N discs to the peg k (k = 0, 1 or 2, and $k \neq i$) using the remaining peg j, such that the following conditions are satisfied:

I. Only one discs may move per each step;
II. Only the top disc from a peg can move at each step;
III. The moving disc can be placed only at an empty peg or on top of a larger disc.

There is a simple algorithm to move N discs from peg i to peg k:

To move N discs from peg i to peg k using peg j {
If N = 0, then do nothing
Otherwise,
Move N – 1 discs from peg i to peg j using peg k;
Print the move from peg i to peg k as a two-letter word with the first letter be the value of i and the second letter – the value of k, and terminate;
Move N – 1 discs from peg j to peg k using peg i.

1. Write a context-free grammar G that generates moves of the Hanoi Tower discs from peg i to peg k according to the outlined algorithm, where i = (sum of your ID# digits) % 3 and $k = (i + 2)$ % 3. Use the alphabet of terminals $\Sigma = \{0, 1, 2\}$. Define the alphabet of non-terminals NT, the set of rules R and the starting state S.
2. Construct a push-down automaton that accepts $L(G)$.
3. Transform the grammar G into the Chomsky Normal Form.

Section 3. Turing Machines (chapter 4)

Task 6. Reconsider cyclic numbers from **Task 3**. Construct a Turing machine that reads from the first cells of the input tape the base digit in base-5 numeral system and computes the cyclic number digit-by-digit starting from the first cell. $\Sigma_5 = \{0, 1, 2, 3, 4\}$. The base-5 multiplication table is given below for your reference:

	0	**1**	**2**	**3**	**4**
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	11	13
3	0	3	11	14	22
4	0	4	13	22	31

Task 7. Consider at the beginning of otherwise empty input tape a 5-symbol string $w \in \{b, d\}^5$ that represents the status of the first 5 cells with *b* corresponding to the born state and *d* – dead state. Construct a single-tape two-head Turing machine that computes the next status of the middle cell based on the current status of its neighbors and itself according to the following rules:

a. The originally dead cell is born, if it has 2 or 3 born neighbors;
b. The already born cell survives, if it has 2 or 4 born neighbors
c. In all other cases the cell dies.

Task 8. Repeat **Task 7** and construct a standard single-tape single-head Turing machine that computes the next status of the middle cell based on the same rules as in **Task 7**.