1. Reading: D. Kozen *Automata and Computability*, Lectures 19,20
J. Hopcroft and J. Ullman *Introduction to Automata Theory, etc.*, section 4.1, 4.2.

2. The main message of this lecture:

> **Finite (memoryless) automata work fine in soda dispensers, pay-phones, photocopying machines, etc. However, they are not capable of handling even the most basic intellectual tasks, e.g. parsing. New devices are needed here: pushdown automata and context-free languages.**

A computer program is usually written as a long string of ASCII characters (a high-level *source program*) and it is compiler's job to read this string and to produce an equivalent low-level *object program*. Finite automata cannot serve as compilers since the former are not able even to determine a syntactic structure of a program (to perform *parsing*). For example, the most trivial parsing problem is to recognize balanced parentheses. By the Pumping Lemma we may fool every finite automaton accepting balanced parentheses strings and show that it in fact accepts many unbalanced strings as well.

Compiling takes computational devices with some special sort of unlimited memory called *pushdown automata* (PDA). The corresponding class of formal languages is called *context-free languages* (CFL).

Example 17.1. Parsing well-parenthesized expressions of propositional logic. There are propositional variables $P, Q, R, \ldots$, boolean constants $\bot, \top$ (for *false* and *true* respectively), binary connectives $\wedge, \vee, \rightarrow, \leftrightarrow$, unary connective $\neg$, as well as parentheses), (. The traditional definition of (well-parenthesized propositional) expressions is

A. variables and constants are expressions,
B. if φ and ψ are expressions, and $*$ a binary connective, than $(\varphi * \psi)$ is again an expression
C. if φ is an expression, than $(\neg\varphi)$ is an expression.
D. There are no expressions other than obtained by A - C.

Here is a typical expression: $(((P \vee Q) \rightarrow R) \leftrightarrow (((\neg P) \vee R) \wedge ((Q \rightarrow \bot) \vee R)))$
and its step-by-step justification:

1. $P, Q, R, \bot$ are expressions, by A
2. $(P \vee Q)$, $(\neg P)$, $(Q \rightarrow \bot)$, by B, C, from 1
3. $((P \vee Q) \rightarrow R)$, $((\neg P) \vee R)$, $((Q \rightarrow \bot) \vee R)$, from 1,2, by B,C
4. $(((\neg P) \vee R) \wedge ((Q \rightarrow \bot) \vee R))$, from 3, by B,C
5. $(((P \vee Q) \rightarrow R) \leftrightarrow (((\neg P) \vee R) \wedge ((Q \rightarrow \bot) \vee R)))$, from 3,4. by B,C

It is an easy exercise to show that the set of strings PROP consisting of all well-parenthesized propositional expressions is not regular. What else to expect from finite automata if they cannot even recognize parentheses correctly!.

Definition 17.2. A context-free grammar (CFA) is $G = (N, \Sigma, P, S)$, where
N is a finite set of *nonterminal symbols*
Σ is a finite set of *terminal symbols* disjoint with N
P is a finite set of *productions* of the format $A \longrightarrow \alpha$, $A \in N$, $\alpha \in (N \cup \Sigma)^*$
$S \in N$ the *start symbol*

Let $\alpha, \beta \in (N \cup \Sigma)^*$. We say that β is derivable from α in one step, notation $\alpha \xrightarrow{1} \beta$, if β can be obtained from α by replacing some occurrence of a nonterminal symbol A in α by γ such that $A \longrightarrow \gamma \in P$. For $n \geq 0$ $\alpha \xrightarrow{n} \beta$ means that β is derivable from α in n steps, $\alpha \xrightarrow{*} \beta$ if $\alpha \xrightarrow{n} \beta$ for some $n \geq 0$. The *language generated* by G is $L(G) = \{x \in \Sigma^* \mid S \xrightarrow{*} x\}$

Example 17.3. Generating $\{a^n b^n \mid n \geq 0\}$: $N = \{S\}$, $\Sigma = \{a, b\}$, $P = \{S \longrightarrow aSb,\ S \longrightarrow \epsilon\}$. Derivation on a string $a^n b^n$ is

$$S \xrightarrow{1} aSb \xrightarrow{1} aaSbb \xrightarrow{1} aaaSbbb \xrightarrow{1} aaaaSbbbb \ldots \xrightarrow{1} a^n S b^n \xrightarrow{1} a^n b^n$$

Convention 17.4. Vertical bar notation: we write $A \longrightarrow \alpha_1 | \alpha_2 | \ldots | \alpha_n$ instead of

$$A \longrightarrow \alpha_1,\ A \longrightarrow \alpha_2,\ \ldots A \longrightarrow \alpha_n$$

Example 17.5. Generating PAREN = set of strings of balanced parentheses: $N = \{S\}$, $\Sigma = \{[,]\}$, $P = \{S \longrightarrow [S] \mid SS \mid \epsilon\}$.

Example 17.6. Generating palindromes over $\Sigma = \{a, b\}$: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \epsilon$

Example 17.7. Generating well-parenthesized propositional expressions: $N = \{S, B, U, C, V\}$,

$S \Rightarrow (SBS) \mid (US) \mid C \mid V$
$B \Rightarrow \vee \mid \wedge \mid \rightarrow \mid \leftrightarrow$
$U \Rightarrow \neg$
$C \Rightarrow \bot \mid \top$
$V \Rightarrow P \mid Q \mid R \mid \ldots$

Homework problems: ♯ 69, 70. p.333 of Kozen's book.