CSE443
Compilers

Dr. Carl Alphonce
alphonce@buffalo.edu
343 Davis Hall

http://www.cse.buffalo.edu/faculty/alphonce/SP17/CSE443/index.php
https://piazza.com/class/iybn4ndqa1s3ei
Syllabus

- Posted on website
- Covered most last class
- Academic Integrity
Departmental Policy on Violations of Academic Integrity (AI)

The CSE Department has a zero-tolerance policy regarding academic integrity (AI) violations.

When there is a potential violation of academic integrity in a course, the course director shall first notify the concerned students. This notification begins the review and appeals process defined in the University's Academic Integrity statement:

http://catalog.buffalo.edu/policies/course/integrity.html

Upon conclusion of the review and appeals process, if the department, school, and university have determined that the student has committed a violation, the following sanctions will be imposed upon the student:

§ 1. Documentation. The department, school, and university will record the student's name in departmental, decanal, and university-level academic integrity violations databases.

§ 2. Penalty Assessment. The standing policy of the Department is that all students involved in an academic integrity violation will receive an F grade for the course. The course director may recommend a lesser penalty for the first instance of academic integrity violation, and the adjudication committees that hear the appeal at the department, decanal and provost level may recommend a lesser or greater penalty.
Phases of a compiler

Figure 1.6, page 5 of text
Lexical Analysis

- Lexical structure described by regular grammar
- Deterministic finite state machine performs analysis
If $L$ and $M$ are regular, so are:

- $L \cup M = \{s | s \in L \text{ or } s \in M\}$  \hspace{1cm} \text{UNION}
- $LM = \{st | s \in L \text{ and } t \in M\}$  \hspace{1cm} \text{CONCATENATION}
- $L^* = \bigcup_{i=0,\infty} L^i$  \hspace{1cm} \text{KLEENE CLOSURE}

By definition, $L^0 = \{\varepsilon\}$
Given an alphabet $\Sigma$

**REGular EXpression (regex)**

**Inductive definition**

- $\varepsilon$ is a regex
  
  $$\mathcal{L}(\varepsilon) = \{\varepsilon\}$$

For each $a \in \Sigma$, $a$ is a regex

$$\mathcal{L}(a) = \{a\}$$
Regular expressions (regex)
Inductive definition

Assume \( r \) and \( s \) are regexes.

\[ r|s \] is a regex denoting \( \mathcal{L}(r) \cup \mathcal{L}(s) \)
\[ rs \] is a regex denoting \( \mathcal{L}(r) \mathcal{L}(s) \)
\[ r^* \] is a regex denoting \( (\mathcal{L}(r))^* \)
\[ (r) \] is a regex denoting \( \mathcal{L}(r) \)

Precedence: Kleene closure > concatenation > union
Associativity: all left-associative (minimize use of parentheses: \( (r|s)|t = r|s|t \) )
Assume \( r \) and \( s \) are regexes.

**Commutativity** \( r|s = s|r \)

**Associativity** \( r|(s|t) = (r|s)|t \) and \( r(st) = (rs)t \)

**Distributivity** \( r(s|t) = rs|rt \) and \( (s|t)r = sr|tr \)

**Identity** \( \varepsilon r = r\varepsilon = r \)

**Idempotency** \( r** = r^* \)
We can describe a regular language using a regular expression
A regular expression can be recognized using a finite state machine.

Machines:
NFA
non-deterministic finite automaton
DFA
deterministic finite automaton
Process of building lexical analyzer

1) spell out the language
Process of building lexical analyzer

2) formulate a regular expression
Process of building lexical analyzer

3) build an NFA
Process of building lexical analyzer

4) transform NFA to DFA
Process of building lexical analyzer

5) transform DFA to a minimal DFA
5) The minimal DFA is our lexical analyzer.

Diagram:
- Language
- Regex
- NFA
- DFA
- Character stream
- Token stream
- DFA
- Lexical analyzer
Focus for today

regex -> NFA
Nondeterministic Finite Automata (NFA)

- A finite set of states $S$
- An alphabet $\Sigma$, $\varepsilon \notin \Sigma$
- $\delta \subseteq S \times (\Sigma \cup \{\varepsilon\}) \times \mathcal{P}(S)$ (transition function)
- $s_0 \in S$ (a single start state)
- $F \subseteq S$ (a set of final or accepting states)
Deterministic Finite Automata (DFA)

- A finite set of states $S$
- An alphabet $\Sigma$, $\varepsilon \not\in \Sigma$
- $\delta \subseteq S \times \Sigma \times S$ (transition function)
- $s_0 \in S$ (a single start state)
- $F \subseteq S$ (a set of final or accepting states)
Regex $\Rightarrow$ NFA

for each $a \in \Sigma$
Regex -> NFA

$S^*$

$N(s)$ $N(t)$ $1$

$S^*$

$N(s)$
Simple example

static
Simple example

static
Simple example

static

struct