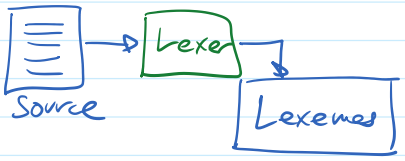
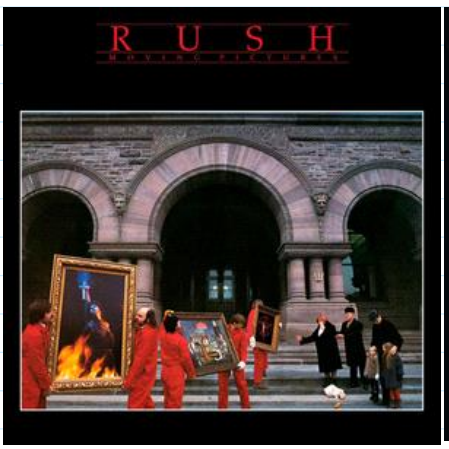
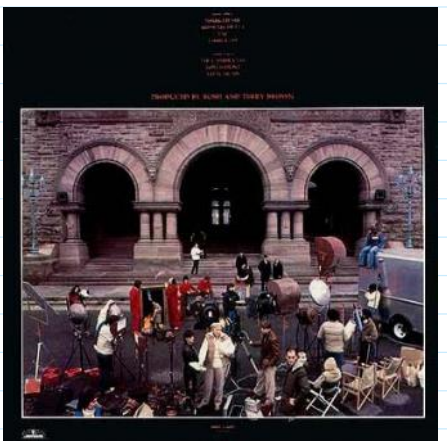
4 Syntax and Regular Expressions

Wednesday, January 31, 2024 11:57 AM



- from Linguistics:
  - Orthography :- How to write words?
    apple manzana pomme ЯБЛО
    given by a Dictionary:
    in a programming language : keywords:
    FOR WHILE FUNCTION.
  - Syntax :- How to write sentences?
    "has jump apple dog green"
  - Semantics :- What do sentences mean?
    "colorless green ideas sleep furiously" N. Chomsky
    Adj Adj noun verb adverb.
    "the chickens are ready to eat"
    "Moving Pictures"





- For a programming language.
  - How to write words?
    - Keywords :- for while if end ...

- How to write words?
  - Keywords :- for while if end ....
  - "identifiers" :- named entity.
  - "Literals" :- numerical values (integer, reals, hexes, scientific)
    strings
    (arrays)
    (dictionaries)
    (tuples)

To Answer This Question:
"Mathematical Languages" S. Kleene E. Post.

Def:- Alphabet :- a finite set of symbols

e.g. $\Sigma = \{a, b\}$

Language :- a set of sequences (words) made of symbols from the alphabet

e.g $L = \{a, aa, aba, baa, bbb\}$

1:- Specification Problem:
How to precisely describe a language?

$L = \{a, ab, aba, abb, ...\}$
???

2:- Recognition Problem
given a word w and a language L
answer $w \in L$?

Answer 1 :- Specification :- Regular Expressions

built from: Σ | * ( )

Regular Expressions:
- a word w is a regular expression
- if r1 and r2 are regular expressions
  - r1 r2
  - r1 | r2

} are also regular expressions

- r1 r2
- r1 | r2
- r1*

} are also regular expressions

A regular expression specifies a language

- w — {w}
- r1 r2 — $\{w_1 w_2 \mid w_1 \in r1 \wedge w_2 \in r2\}$
- r1 | r2 — $\{w \mid w \in r1 \vee w \in r2\}$
- r* — {w | w is zero or more repetitions of a word in r}

r*: Kleene star.

Eg: (a | b) b = {ab, bb}

a : {a}
b : {b}
a|b : {a, b}

Eg: a(b*) = {a, ab, abb, abbb, abbbb, ...}

Specifying an infinite set through a finite expression.

Eg: a* = {ε, a, aa, aaa, aaaa, aaaaa, ...}

ε: empty string.

Eg: (aa | b*) a = {a, ba, aaa, bba, bbba, bbbba, ...}

{aa} {ε, b, bb, ...}

- Shorthands

r? ≡ (r | ε) "to r or not to r"

r+ ≡ r(r*) "one or more repetitions of a word in r"

- Regular Expressions AND Programming Languages

Σ = ASCII keyboard characters.

NOTE:- we cannot use regular expressions to describe a programming language.

$\mathcal{L}_{C++}$ = The set of all valid C++ programs.

Why?

Consider $\{a^n b^n \mid n \in \mathbb{Z}^+\} = \{ab, aabb, aaabbb, aaaabbbb, \ldots\}$

like matching brackets

{ ( ) [ ] } note: counting is not enough.

{ ([ )] }

Famous Stack Overflow Question:

"Regular Expression to validate HTML?" Closed

<a> ...
<h3> ... </h3>

- RegEx: are good enough to describe "tokens"

  token:- a category of atomic strings.
  lexeme:- an instance of a token.

  - "identifiers":- named entity.
  - "Literals":- numerical values, strings. (integer, reals, hexes, scientific)

while (flag == 62.3 else if) 14 7 != x + var_1

Keyword, ident, op, real literal, keyword, int Lit, op, ident.

- Describing tokens using RegEx

  Σ = keyboard characters

  - Shorthands.

    $[0\text{-}9] \equiv (0 \mid 1 \mid 2 \mid 3 \mid \cdots \mid 8 \mid 9)$

    $[A\text{-}Z] \equiv (A \mid B \mid C \mid \cdots \mid Y \mid Z)$

$[0-9] \equiv (0 \mid 1 \mid 2 \mid 3 \mid \cdots \mid 8 \mid 9)$

$[A-Z] \equiv (A \mid B \mid C \mid \cdots \mid Y \mid Z)$

$[a-z] \equiv (a \mid b \mid c \mid \cdots \mid y \mid z)$

$[A-Za-z] \equiv ([A-Z] \mid [a-z])$

- Regex for keywords.

  e.g C++ (if | while | for | do | else | .....)

- Regex for integer literals

  $(+|-)?([0-9])^+$ { 0, 1, 123, 001, 008, −5, +27, −001, +001 }

- Regex for identifiers.

  Fortran $([A-Z])([A-Z] \mid [0-9])^*$ A, A2, M7, APPLE3

  Pascal $([A-Za-z])([A-Za-z] \mid [0-9] \mid \_)^*$ var_1, point_3D, initX

  Python $([A-Za-z] \mid \_)([A-Za-z] \mid [0-9] \mid \_)^*$ var_1, this_is_snake, __init__, __main__, ___, _____, _____

- Regex for scientific notation.

  eg. $3.7e^{-5}$ $-8.9e^{7}$ $123e^{3}$

$$(+|-)?\,[0-9]^+\,(\,.\,[0-9]^+)?\,\text{`e'}\,(+|-)?\,[0-9]^+$$

- Regex can also be used to describe other standard formats.
  - phone numbers
  - SSN
  - URL's
  - Serial numbers
  - licence plates
  - etc.....

—o— EOF