به نام دانای نیآموخته

درس: کامپایلر

ترم: نیمسال اول(۰۴-۰۵)

دپارتمان: مهندسی کامپیوتر

مدرس: دکتر شیما شفیعی

Programming Language Specs

- Since the 1960s, the syntax of every significant programming language has been specified by a formal grammar
  - First done in 1959 with BNF (Backus-Naur Form), used to specify ALGOL 60 syntax
  - Borrowed from the linguistics community (Chomsky)

Positive integer

```
<positive-integer> ::= <non-zero-digit> <digits>
<digits>           ::= ε | <digit> <digits>
<digit>            ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
<non-zero-digit>   ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

(2+3)* 4+5

```
<expr>   ::= <term> | <term> + <expr>
<term>   ::= <factor> | <factor> * <term>
<factor> ::= ( <expr> ) | <number>
<number> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

number> → 2 , 3 , 4 , 5 >

factor> → (2 + 3) , 4 , 5 >

term> → (2 + 3) * 4 >

expr> → (2 + 3) * 4 + 5 >

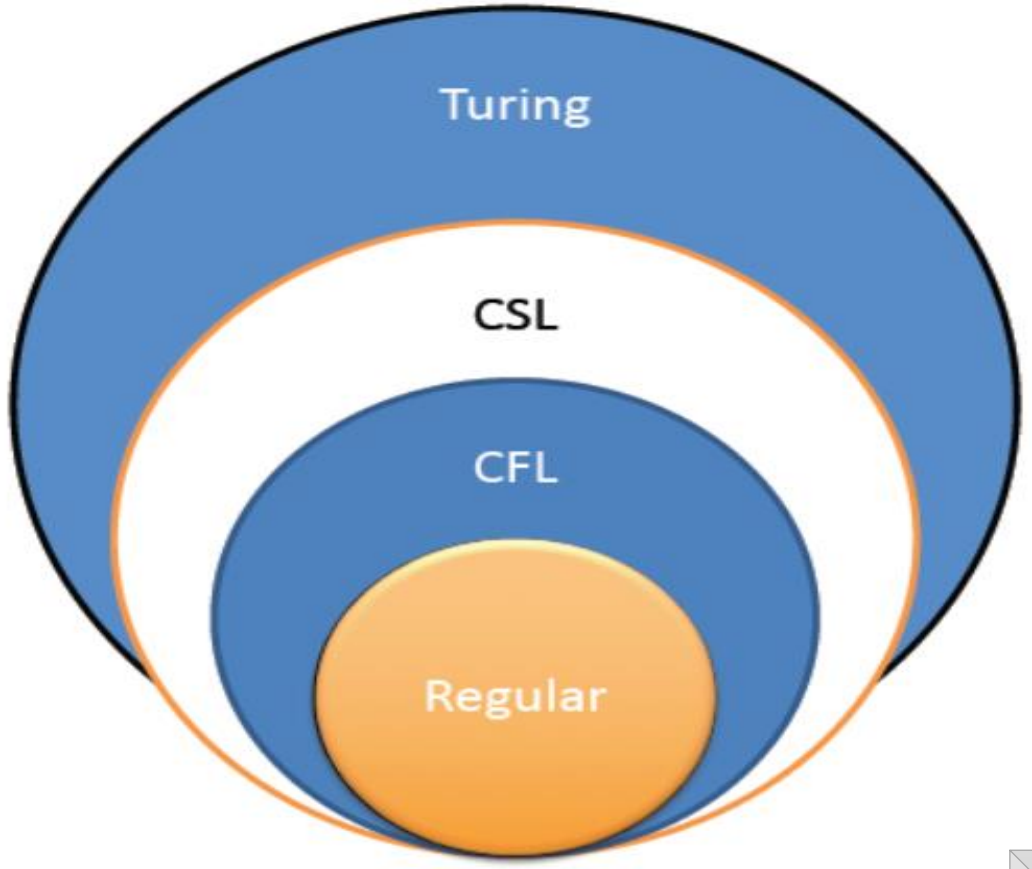
- <expr>عبارات ریاضی‌ای را تولید می‌کند که شامل + هستند.
- <term>بخش‌هایی از عبارت هستند که شامل * هستند.
- <factor>می‌تواند یک عدد یا یک عبارت داخل پرانتز باشد.
- <number>ارقام ۰ تا ۹ هستند.

Formal Languages & Automata Theory (a review on one slide)

- Alphabet: a finite set of symbols and characters
- String: a finite, possibly empty sequence of symbols from an alphabet
- Language: a set of strings (possibly empty or infinite)
- Finite specifications of (possibly infinite) languages
  - Automaton – a recognizer; a machine that accepts all strings in a language (and rejects all other strings)
  - Grammar – a generator; a system for producing all strings in the language (and no other strings)
- A particular language may be specified by many different grammars and automata
- A grammar or automaton specifies only one language

Language (Chomsky) hierarchy: quick reminder

- Regular (Type-3) languages are specified by regular expressions/grammars and finite automata (FSAs)
  - Specs and implementation of scanners
- Context-free (Type-2) languages are specified by context-free grammars and pushdown automata (PDAs)
  - Specs and implementation of parsers
- Context-sensitive (Type-1) languages … aren't too important (at least for us)
- Recursively-enumerable (Type-0) languages are specified by general grammars and Turing machines



program ::= *statement* | *program statement*

statement ::= *assignStmt* | *ifStmt*

assignStmt ::= *id* = *expr* ;

ifStmt ::= if (*expr*) *statement*

expr ::= *id* | *int* | *expr* + *expr*

id ::= a | b | c | i | j | k | n | x | y | z

int ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Grammar Rule	Example
program ::= statement	a = 5 ;
program ::= program statement	a = 5 ; if (a) b = 3 ;
statement ::= assignStmt	x = 4 ;
statement ::= ifStmt	if (1) y = 2 ;
assignStmt ::= id = expr ;	z = a + 1 ;
ifStmt ::= if (expr) statement	if (b) k = 2 ;
expr ::= id	c
expr ::= int	7
expr ::= expr + expr	i + 3
id ::= a	a
int ::= 1	1

Exercise: Derive a simple program

program ::= *statement* | *program statement*
statement ::= *assignStmt* | *ifStmt*
assignStmt ::= *id* = *expr* ;
ifStmt ::= if (*expr*) *statement*
expr ::= *id* | *int* | *expr* + *expr*
id ::= a | b | c | i | j | k | n | x | y | z
int ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

a = 1 ; if (a + 1) b = 2 ;

Productions

- The rules of a grammar are called productions
- Rules contain:
  - Nonterminal symbols: grammar variables (program, statement, id, etc.)
  - Terminal symbols: concrete syntax that appears in programs (a, b, c, 0, 1, if, =, (,), …
- Meaning of:

  nonterminal ::= <sequence of terminals and nonterminals>

  In a derivation, an instance of nonterminal can be replaced by the sequence of terminals and nonterminals on the right of the production
- Often there are several productions for a nonterminal – can choose any in different parts of derivation

گرامر برای برنامه

```
a = 1 ;
```

```
program     ::= statement
statement   ::= assignStmt
assignStmt  ::= id = expr ;
expr        ::= int
id          ::= a
int         ::= 1
```

گرامر برای برنامه

```
a = 1 ; if ( a + 1 ) b = 0 ;
```

```
program     ::= statement | program statement
statement   ::= assignStmt | ifStmt
assignStmt  ::= id = expr ;
ifStmt      ::= if ( expr ) statement
expr        ::= id | int | expr + expr
id          ::= a | b
int         ::= 0 | 1
```

تمرین ۵

- گرامر برای دو برنامه آورده شده نیز بنویسید.

- if (x+1)y=0;
- if (y+0); if (x)x=1;

مهلت تحویل: ۲۵ مهرماه ۱۴۰۴

موفق باشید