

## Module C Evaluation Workshop

- Fri., Nov. 14, class time (one week from today)
- Review the relevant part of the syllabus and the on-line handbook
- Re-use your manila folder or large envelope; **Include Module B**
- Complete and bring "Take-Home Exercise Self-Evaluation Form" (distributed today) along with exercises
- Exercise C5 will be checked on Tue. Solutions available that day.
- Complete and bring "Module C Comprehensive Exercises" (available on-line)
- Group evaluation sessions (open book): 20 min × 3
- "Comprehensive Exercise Self-Evaluation Forms" will be distributed that day (no need to print in advance) **New criteria; Preview on-line**
- Submit all materials at the end of the session
- Mini Project Phase 2 **due on Tue., Nov. 18** (separate)

CMSC210 C5

1

Read the mini project page

## Mini Project Phase 2

- Respond to the feedback of the instructor on Phase 1, if any. Especially, the **logic-structure connection**.
- Formally define the structure of your object/phenomenon. Also try to define the structure components as much as possible
- Write up your logical specification as clearly as possible. Try to use First-Order Logic formulae (not required).
- Explain how the structure would satisfy the logical statements. Examine whether unintended structures would also satisfy the logical statements.
- There is no length requirement. Your mini project must be word-processed (except for special symbols/diagrams/schematics, if any).
- Self-evaluation (must be included at the end of the submission): Make sure that your submission satisfies the above requirements. Give 1 pt if it is the case.

CMSC210 C5

2

## Java Comments (Comprehensive A)

- A standard C-style comment, where all of the characters between `/*` and `*/` are ignored.
- A collection including `/**`, `/* */`, `/*a*/`, `/*b*/`, etc.

Today's Summary Exercise

CMSC210 C5

3

## Compilers

- High-level language → Machine language  
(source language) (target language)
- **Specification** of the source language
- **Mechanism** of analyzing and translating the source language

CMSC210 C5

4

## Floating-Point (FP) Numbers

- Examples: 1.5, 345, 0 **cf. 0.1.2, \$19.95+s/h**
- Specification
  - **Integer part**: either a 0 or a non-zero number followed by any number of digits
  - **Fraction part (optional)**: '.' and a digit followed by any number of digits
- **Mechanism**: Must be able to handle possible repetition of any length

CMSC210 C5

5

## Program Structure

- Example:

```
if (x>0) {
    if (x<0) {doThis();}
    else {doThat();}
}
else {doWhat();}
```
- Specification: Must be able to represent parenthesis matching
- Mechanism: Must be able to handle parenthesis matching

CMSC210 C5

6

## Example: Virtual Pet (Ex-B3)

		Stimuli			
		Bison ( <i>b</i> )	Geisha ( <i>g</i> )	Wormhole ( <i>w</i> )	
States	Puzzled ( <i>p</i> )	Puzzled ( <i>p</i> )	Happy ( <i>h</i> )	Mad ( <i>m</i> )	
	Happy ( <i>h</i> )	Mad ( <i>m</i> )	Puzzled ( <i>p</i> )	Sleepy ( <i>s</i> )	
	Mad ( <i>m</i> )	Happy ( <i>h</i> )	Mad ( <i>m</i> )	Puzzled ( <i>p</i> )	
	Sleepy ( <i>s</i> )		Sleepy ( <i>s</i> )		

CMSC210 C5

7

## C5: Languages/Automata

### Today

- Specify and process certain types of sets
  - Languages, automata, and grammars general
  - Regular languages, finite-state automata, and regular expressions special case
- Take-home exercises
  - Binary numbers, English spelling, Musical code (sample solutions to be posted on Tue)

CMSC210 C5

8

### Section 1

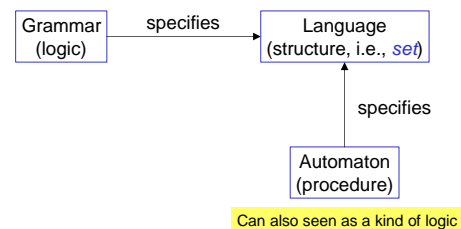
## Three Aspects

- Language:** A set of strings
- Automaton:** A machine to process a language [plural: automata]
- Grammar:** Specification of a language
- Example
  - The English language (set of sentences), described by the English grammar, spoken by a person (automaton)

CMSC210 C5

9

## Schematically,



CMSC210 C5

10

## Floating-Point (FP) Numbers

- Example: 1.5, 345, 0
- Language:** The set of all FP numbers
- Grammar**
  - Integer part:** either a 0 or a non-zero number followed by any number of digits
  - Fraction part (optional):** '.' and a digit followed by any number of digits
- Automaton:** Some machine that can handle possible repetition of any length

CMSC210 C5

11

## Program Structure

- Example:
 

```

            if (x>0) {
              if (x<0) {doThis();}
              else {doThat();}
            }
            else {doWhat();}
            
```
- Language:** The set of valid programs
- Grammar:** Program specification
- Automaton:** Some machine that can handle parenthesis matching As simple as FP?

CMSC210 C5

12

## Section Summary

- Language = \_\_\_\_\_ of strings
- Grammar = specification
- Automata = machine (also as specification)
- Languages can be characterized by automata and grammars.
- There are different types of languages/automata/grammars, appropriate for different tasks.

CMSC210 C5

13

## Group Exercise 1

- FOL can be viewed in terms of the connection between language, grammar, and automaton. Complete the following:
  - Language: The set of \_\_\_\_\_
  - Grammar: Choice of symbols and rules to represent \_\_\_\_\_
  - Automaton: Some machine that \_\_\_\_\_

Note: Different perspective

- FOL (specifically chosen part) used to specify a structure
- FOL (as a whole) being specified as a set

CMSC210 C5

14

## Section 2

### Language of FP (and the like)

- A subset of all strings
    - Language = structure consisting only of a single set
    - For processing FP, we should not consider arbitrary concatenation (unlike the **String** structure in C1). I.e., not closed under concatenation
- E.g.,  $19.95 + 6.95 = 19.956.95$

CMSC210 C5

15

### Preliminary Specification

- Define
  - $Z = \{0\}$
  - $N = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  - $D = Z \dot{\cup} N$
  - $P = \{', '\}$
- Integer part:  $Z$  or  $N$  or  $ND$  or  $ND \dots$
- Fraction part (optional):  $PD$  or  $PDD \dots$

Precise, yet compact representation?

CMSC210 C5

16

## Regular Expression

- Specification (grammar) based on
  - Sequence:  $XY$       Notation:  $X^n = \underbrace{X \dots X}_n$
  - Alternative:  $X | Y$
  - Repetition (zero or more times):  $X^*$
- Integer part:  $Z | (ND^*)$
- Fraction part (optional):  $PDD^*$
- Combination:  $(Z | (ND^*)) (\emptyset | (PDD^*))$

A single regular expression represents a set.

CMSC210 C5

17

## Group Exercise 2

- Variable name requirements in some programming language:
  - Must begin with an alphabetic character
  - Must consist of alphanumeric characters
- Give a regular expression that would specify the language (set of variable names)
  - $A = \{a, b, \dots, z, A, B, \dots, Z\}$
  - $N = \{0, 1, \dots, 9\}$

CMSC210 C5

18

## More Examples

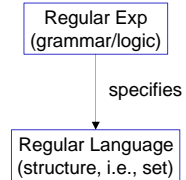
- $(a|b|v|r|d|e|z|z|n|k|l|m|n|o|p|c|t|y|f|x|u|v|w|z|y|k|l|o|y|' )^*$  abuse of notation
- $S^*$  where  $\Sigma = \{a, b, \dots, z\}$  Seen before?
- $(\bullet - | - \bullet \bullet \bullet | - \bullet - \bullet | \dots | - - \bullet \bullet | pause)^*$
- $((\emptyset | \text{anti} | \text{non-})\text{symmetric}) | (((\emptyset | \text{non-})\text{transit}) | ((\emptyset | \text{ir} | \text{non-})\text{reflex}) \text{ive})$
- $CV(CV)^*$  where  $V = \{a, e, i, o, u\}$ ,  $C = \{a, b, \dots, z\} - V$
- $a^{2n}$  ( $n$  is a natural number) Corresponding RegExp?

CMSC210 C5

19

## Regular Language

- A language that can be specified by a regular expression (also called **regular set**)
- Examples
  - Floating-point numbers
  - Variable names
  - Valid file names
  - Good passwords



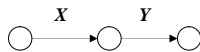
CMSC210 C5

20

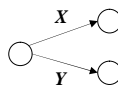
## Processing Regular Languages

- Need to be able to handle

- Sequence



- Alternative



- Repetition (zero or more times)



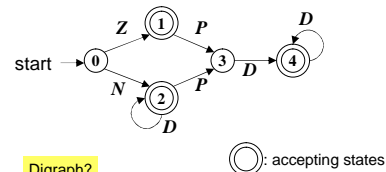
one or more repetition?

CMSC210 C5

21

## Finite-State Automata (FSA)

- A machine that consists of the three components shown on the previous slide
- Example



Digraph?

⊙: accepting states

CMSC210 C5

22

## Examples

- $a^{2n}$  ( $n$  is a natural number) =  $(aa)^*$
- $CV(CV)^*$  where  $V = \{a, e, i, o, u\}$ ,  $C = \{a, b, \dots, z\} - V$
- A vending machine (M&M: \$0.20)
- Digital watch move on nothing
- *This is (∅ / not) a (very)\* long sentence.*
- $((\emptyset | \text{anti} | \text{non-})\text{symmetric}) | (((\emptyset | \text{non-})\text{transit}) | ((\emptyset | \text{ir} | \text{non-})\text{reflex}) \text{ive})$

CMSC210 C5

23

multiple possible moves

## Group Exercise 3

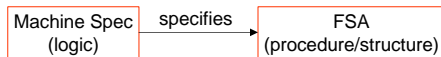
- Assume that all Italian words end in  $a$  or  $o$ .
- Create a FSA that would consider any alphabetic sequence ending in  $a$  or  $o$  as Italian words.
  - Start the FSA at State 0
  - Create two distinct accepting states ⊙, one for  $a$  and another for  $o$ . Use these state to identify masculine vs. feminine gender of the input word.
  - Use  $X$  to denote  $\{a, b, \dots, z\} - \{a, o\}$

CMSC210 C5

24

## FSA as a Structure

- **FSA** =  $(S, I, s_0, F, T)$ 
  - $S$ : Set of states
  - $I$ : Set of input symbols
  - $s_0$ : the initial state,  $s_0 \in S$  (as a constant, 0-ary function)
  - $F$ : Set of accepting (final) states  $F \subseteq S$
  - $T$ : Transition function:  $S \times I \rightarrow S$

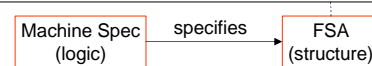
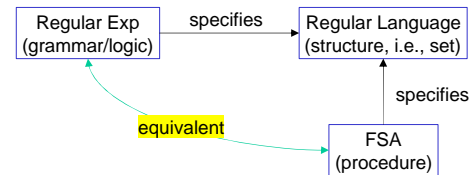


CMSC210 C5

25

## Section Summary

Special case of language-grammar-automaton connection



CMSC210 C5

26

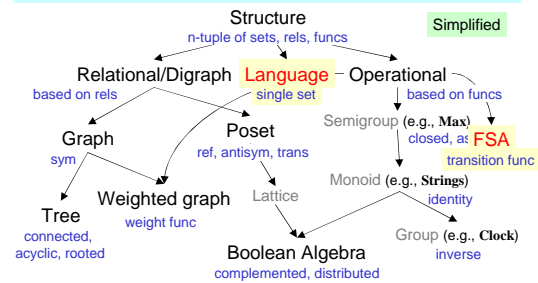
## Applications

- Compilers
  - State analysis (virtual pet)
  - Valid file names
  - Date/time format
  - Password screening
  - URL
  - Unix command line
  - Spell checker
- Recognition
  - Validation
  - Error detection
  - Generation
  - Translation

CMSC210 C5

27

## Organization of Structures



CMSC210 C5

28

## Summary Exercise

- [challenging] Specify the traditional Java comments using either a regular expression or a FSA.
  - You can stop at some point and hand in your ideas.
- Questions/Comments/Suggestions

CMSC210 C5

29