CS 461

Programming Language Concepts

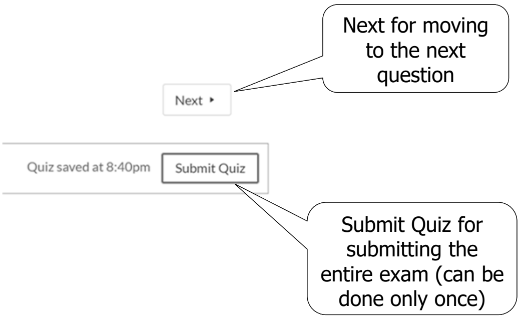
Gary Tan
Computer Science and Engineering
Penn State University

•1

Format

- Exam Time
  - Feb 16th, 7-8pm US Eastern Time (60 minutes)
- Canvas-based exam
  - You will be shown one question at a time
  - Can go forward/backward to next/previous question
  - IMPORTANT: you can submit your exam only once
    - DO NOT PRESS the "SUBMIT QUIZ" button until you are done

•2



Next for moving to the next question

Submit Quiz for submitting the entire exam (can be done only once)

•3

Format

- 60-min exam
  - Closed book and notes
  - Not allowed to communicate with other students during the exam
  - Questions during the exam can be posted to campuswire
    - make them viewable only to instructors and TAs
    - phrase the question to make it easily understandable
- Feb 16th class cancelled
  - Instead, I will move my office hours this week to the 16th class time: 12:20 to 2:20pm

•4

1ST EXAM REVIEW

•5

Types of Questions

- True/false questions
- Multiple-choice questions
- Short answer questions
- Versions of homework problems
- You will be expected to be able to read and write small programs

•6

Ch1 Introduction

- Language = syntax + semantics + philosophy
- Programming paradigms
  - Imperative programming, OO, functional programming, logic programming
- Possible kinds of questions
  - True/false questions; multiple-choice questions
- You do not need to know
  - Date when a language was designed

•7

Ch2 Grammar

- BNF grammar (CFG)
  - Terminals, nonterminals, production rules, start symbol
  - Derivation, left-most vs. right-most derivations, parse tree
- Ambiguity
  - Definition
  - Removing operator ambiguity
    - Adding explicit parenthesis
    - Design a new grammar to enforce associativity and precedence rules
    - Ambiguous grammar + informal spec. of associativity and precedence rules

•8

Ch2 Grammar

- E-BNF
  - Writing more concise grammars
  - Alternatives, repetitions, optional parts
  - E-BNF is no more expressive than BNF
    - Can always convert E-BNF to BNF
- Possible questions
  - Given a grammar, write derivations and parse trees
  - Conversion from E-BNF to BNF
  - Design new BNF grammars
    - E.g., given associativity and precedence
  - Decide whether a grammar is ambiguous and explain why
  - Remove ambiguity

•9

Ch2 Lexical and Syntactic Analysis

- Goal: algorithms for constructing a parse tree from input based on grammars
- Lexical vs. syntactic analysis
  - Lexical analysis: seq of chars to seq of tokens; guided by regular expressions
  - syntactic analysis: seq of tokens to parse trees; guided by CFG

•10

Ch2 Lexical and Syntactic Analysis

- Regular expressions
  - epsilon, a, rs, r|s, r*
  - extended regular expressions
    - r+, r?, [a-z]
- FSA
  - states, input alphabet, transition function
  - accepting an input
  - deterministic FSA vs. nondeterministic FSA
  - Theorem: each RE corresponds to a deterministic FSA
- The construction of lexers
  - a single FSA for all tokens; nextToken()
- The construction of parsers: recursive-descent parsing
  - Limitation: left recursion
  - Left-recursion removal
- Lexer and parser generators

•11

Ch2 Lexical and Syntactic Analysis

- Possible kinds of questions
  - Write regular expressions for syntax of tokens
  - Given a regular expression, develop an FSA/DFA
  - Conversion form extended regular expressions to regular expressions
  - Write pseudo code for recursive descent parsers, given a grammar
  - Left-recursion removal

•12

Ch3 Names, Scopes and Bindings

- Syntactic issues for Naming
  - lexical rules
- Binding: compile time vs. runtime
- Variable
  - common bindings; naming convention
  - l-values vs. r-values
- Scope: decides when a name is visible
  - nested scopes
  - "holes" in scopes
  - scope not the same as lifetime
- Constructs that can introduce a scope
  - Blocks, functions, for-loops, classes, packages (module), namespace

•13

Ch3 Names, Scopes and Bindings

- Static scoping vs. dynamic scoping
  - Algorithm based on stack of dictionaries
- Possible kinds of questions
  - Scoping

•14

Comparing the syntax of E-BNF and Regexps

- They have similar concepts but with different syntax
- E-BNF
  - Alternative parts in parentheses and separated with vertical bars
    - <exp> -> <exp> (+ | -) <exp>
  - 0-or-more repetitions in curly braces { }
    - <num> -> <digit> {<digit>}
  - Optional parts in square brackets [ ]
    - <if-stmt> -> if <test> then <stmt> [else <stmt>]

•15

Comparing the syntax of E-BNF and Regexps

- Regexps
  - Alternation: r1 | r2
  - 0-or-more repetition: r*
  - Optional: r?
  - …

•16