CS 461

Programming Language Concepts

Gang Tan
Computer Science and Engineering
Penn State University

2ND EXAM REVIEW

Format

- Exam Time and Place
  - Apr 4th, 12:20-1:10am (50 minutes)
  - Wartick Lab 110 (this room)
- In-class 50-min exam
  - Closed book and notes
  - All calculators, cell phones, and portable audio players must be put away for the duration of the test
- Office hours moved this week
  - Tu 9-11am Westgate W358
- TA's office hours
  - Tu 10-12pm W328
  - Tu 12-2pm W375

Types of Questions

- True/false
- Choice questions
- Short answer questions
- You will be expected to be able to read and write small **Scheme**, Java, and C programs

Ch9 Functions

- Basic Terminology
  - Functions, procedures, methods, subroutines
  - Arguments vs. parameters
- Parameter passing mechanisms
  - By value, by result, by value result, by ref, by name
  - Pass by value-result vs. pass by reference
    - Alias
  - Parameter passing in major languages (C, C++, Java, Ada, …)
- Implementation of functions calls and returns
  - Stack of activation records
  - An activation record: parameters and local variables, return address, dynamic/static link, return value, saved registers, …
  - Support recursive functions

Ch9 Functions

- Possible questions
  - Figure out the result of a program if a particular parameter passing mechanism is used
  - Be able to draw or interpret a stack of activation records
- You do not need to know
  - The ordering of components in an activation record

Ch11 Functional Programming

- Scheme
  - (E1 E2 … En)
  - (define name …)
    - can be a function value (lambda (x) …)
  - Anonymous functions
  - (if P E1 E2)
  - (cond (P1 E_1) … (Pn E_n) (else E_{n+1}))
  - (let ((x1 E1) (x2 E2) … (xk Ek)) E)
  - (let* ((x1 E1) (x2 E2) … (xk Ek)) E)
  - (letrec ((x1 E1) (x2 E2) … (xk Ek)) E)
  - High-order functions

Ch11 Functional Programming

- Scheme, cont'd
  - List processing: null?, car, cdr, cons
  - Manipulating lists: length, map, reduce, …
    - case analysis and recursion
  - Association lists
  - Currying/uncurrying
- Possible questions
  - Scheme
    - Programming questions
    - Pay attention to parentheses!
    - Don't forget the empty-list case

Lambda Calculus

- Syntax: t ::= x | λx. t1 | t1 t2
  - Parsing convention
- Bound and free variables
- Formal definition of FV(t)
  - Alpha renaming
- Substitution: [t/x] t'
  - May need to rename bound variables
- Reduction rules
  - Beta reduction
  - (λx. t') t → [t/x] t'
  - Normal form

Lambda Calculus

- Programming in lambda calculus
  - Encoding let x=M in N as (lambda x. N) M
  - Booleans: true, false, and, or, if
  - Numbers: 0,1,2, …, plus one, plus

Lambda Calculus

- Possible questions
  - calculate free variables
  - perform substitution
  - perform reduction until getting a normal form
  - encoding boolean and arithmetic functions