CS 461

Programming Language Concepts

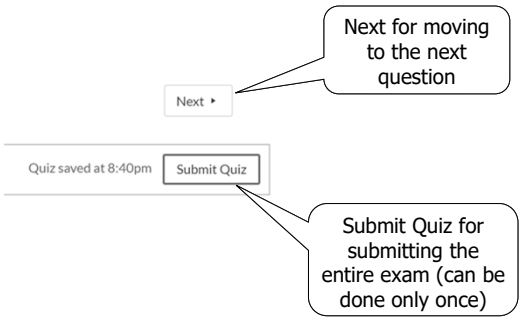
Gary Tan
Computer Science and Engineering
Penn State University

•1

Format

- Exam Time
  - May 2nd, 6:50pm-8:20pm (90 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



•3

Format

- 90-min exam
  - Closed book and notes
  - Not allowed to communicate with anyone other than the instructor/TAs 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
  - General announcements will also be made on campuswire

•4

Format

- Covers everything we have taught in the semester; biased toward material after the second exam
- Extra office hours: May 2nd (Monday) 12:30pm to 2:30pm
  - Using our office hour zoom

•5

Types of Questions

- True/false
- Multiple-choice questions
- Short answer questions
- Versions of in-class exercise/homework problems
- You will be expected to be able to read and write small Racket, Java, and C programs
- Note: Reviews for the other two exams are in Canvas

•6

Lambda Calculus

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

•7

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

•8

Lambda Calculus

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

•9

Purity and Side Effects

- Functional programming vs. imperative programming
  - Pure functional programming: no side effects
  - Imperative programming: mutating the state
- Purity gives you referential transparency
  - Calling the same function with the same input produce the same result
  - Does not depend on where the function call is
  - Enable advanced techniques such as memoization
- Racket constructs with side effects: set!, mcons, set-mcar!, set-mcdr!
- Possible questions: true/false, multiple-choice questions

•10

Ch7 Types

- Types
  - A collection of values and a set of operations on those values
- Type errors: undefined operations
- Static vs. dynamic typing
- Type safety
- Racket is dynamically typed
- Possible kinds of questions
  - Yes/no questions, multiple-choice questions

•11

Memory Management

- The heap: memory manager
- Two routines: p=new(n); delete(p)
- Two ways of mem management in PLs
  - Manual memory management
  - GC
- Racket's way of memory allocation
  - cons and atom cells
- Def. of garbage
- GC algorithms: mark-and-sweep; reference counting; copy collection
  - How does each one work
  - Strengths and weaknesses

•12

Memory Management: Possible Questions

- Figuring out the memory layout after running a program (e.g., Racket)
- Figure out the result after running GC

•13

Ch10 OO Programming

- OO program: a collection of objects that communicate by message passing
  - objects; class
- OO concepts
  - Dynamic lookup
  - Method overloading
  - Subclassing
  - Inheritance
  - Method overriding

•14

OO Programming: Possible Questions

- Figure out results of programs, which may contain OO features:
  - Dynamic lookup; method overloading; method overriding; subclassing; inheritance; class attributes; data attributes; …
- Implementation of dynamic lookup via vtables

•15

SOME OO EXERCISE

•16

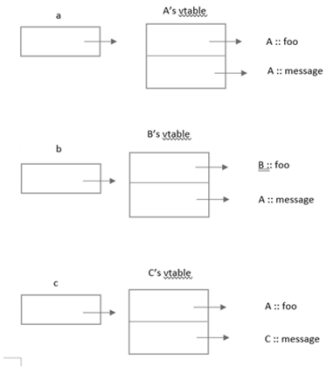
Exercise: Drawing v-tables

```
class A {
  public int foo () { return 1;}
  public void message () { System.out.println( "A"); }
}
class B extends A {   public int foo() {return 2;} }
class C extends A { public void message () { System.out.println( "C");
} }
A a = new A();
B b = new B();
C c = new C();
```

Draw objects a, b, c and their vtables (virtual tables) in the style of the textbook. See Fig. 10.2 and 10.3 of the fourth edition of the book by Scott

•17

Exercise: Drawing v-tables



•18