(While studying all of the material below would be helpful, I realize that not everyone has infinite time, so I have marked with an asterisk those slide sets and labs that are likely to be most useful in preparing for the exam.)

**Slide sets:**

RSS

\*NaturalNumber

\*References

\*Arrays and References

\*Contracts

Repeated Arguments

Hints on Interval Halving

\*Mathematical String Notation

\*Recursion: Thinking About It

\*Concepts of Object-Oriented Programming

\*Static methods vs. instance methods

Recursion: Why it Works

\*Recursion on Trees

\*Testing

JUnit

**Labs:**

XMLTree RSS Processing

Debugging

\*Tracing Parameter Passing

Integer Root with Interval Halving

\*Recursion on NaturalNumber – Static Methods

\*Arrays and References

\*Recursion on NaturalNumber – Instance Methods

\*Recursion on XMLTree

Getting Started with JUnit

**Terms/Concepts:**

RSS XML (XML) element (XML) attribute

Debugging overflow kernel (methods)

Single point of control over change mathematical model

Constructor receiver distinguished formal parameter

Reference type primitive type mutable

Immutable # (in a contract) reference value

Object value memory address assignment (prim. Vs. ref.)

Anonymous variable uninitialized variable garbage collector

Alias parameter passing (prim. Vs. ref.)

Equality checking (prim. Vs. ref.) swapping

Package parameter modes (restores, clears, updates, replaces)

Repeated arguments (math) floor (math) ceiling

(math) string empty string concatenation

Substring prefix suffix

Recursion tracing with recursion fast powering

Sub/derived/child interface (or class) super/base/parent interface (or class)

Overriding overloading declared/static type

Object/dynamic type interface type class type

Polymorphism mathematical induction expression trees

Operand operator testing unit testing

Integration testing system testing method correctness

Test case test plan/fixture boundary case

Routine case challenging case JUnit framework

**OSU components:**

SimpleReader

SimpleWriter

XMLTree

NaturalNumber

Standard

**Java code:**

long interface class extends implements

this java.util.Arrays.deepEquals null

NullPointerException super(…) @Override @Test

static import static assertEquals(…) assertTrue(…)

**Concepts related to recursion:**

The first concept related to recursion

**Remaining concepts related to recursion:**

The first remaining concept related to recursion

**The rest of the remaining concepts related to recursion:**

(you get the idea)