# CS211 About Prelim 1

**The prelim is**
**7:30-9:00PM, Thursday, 11 March 2001, Uris Auditorium**

**The only reason for not taking it then is that you have a conflict with another evening prelim.**
**If this is the case, please let Gries know by Sunday evening, 7 March.**

**There will be a review session on Sunday, 7 March, 3-5PM, Upson B17**

We will try to make the prelim reasonable in length. Someone who knows the material well should be able to complete the prelim in an hour.

Prelim I will cover the general topics of (1) recursion, (2) classes and subclasses, (3) abstract classes, (4) exception handling, and (5) interfaces, (6) nested classes, (7) algorithms linear search, binary search, and partition. To study for the prelim, review the handouts provided for the lectures and the appropriate pages in the text (Weiss or ProgramLive). Also, learn definitions and practice writing programs (or program segments). Hashing will not be covered on this prelim; it will be covered on the second prelim.

The course webpage contains prelims for the previous five semesters, with sample answers.

**1. Recursion.** Know the definition of (1) a recursive definition, (2) a recursive method, (3) base case, recursive case. Know our four steps in understanding a recursive method (see powerpoint slide 7 of the handout on recursion for lectures 3 and 4). Know and be able to carry out the steps in executing a method call. You will be asked to write at least one recursive method on the prelim, and you may be asked to execute a method call.

**2. Classes and subclasses.** The handout for the lecture that reviewed classes as well as the Java bootcamp contains the important material that you must know: (1) difference between public and private; (2) the class as a file drawer (what goes in it?); (3) the format of an instance of class, as a manila folder; (4) the frame for a method call (don't worry about the scope box); (5) the steps in executing a method call; (6) how to evaluated a new expression; (7) the format for an instance of a subclass; (8) overriding a method; (9) explicit and implicit casting and the apparent and real types of a variable; (10) class Object and its methods equals and toString. Also: (11) writing constructors in a subclass and (12) use of method toString.

On the prelim, you may be asked to write a class or subclass. You may be asked to evaluate a new expression. Practice these things.

**3. Abstract classses.** Recitation 3 talked about abstract classes. You should know how to make a class abstract and the consequences of doing so. You should know how to make a method abstract and the consequences of doing so.

**4. Exception handling**. Know: (1) how to write an Exception; (2) how to throw an Exception; (3) how to catch an Exception using a try statement; (4) the rules for propagating a thrown Exception or Error.

**5. Interfaces**. Know: (1) how to write an interface; (2) how to implement an interface and what that entails; (3) casting to and from interfaces; the class-interface hierarchy (see slides 13-15 of the handout for lecture 08).

You should know interface Comparable (see slide 3 of the handout for lecture 08) and be able to write a method that uses it (e.g. in a method that sorts an array whose base type implements class Comparable). You may be asked write an interface or to implement an interface.

**6. Nested classes**. See slides for lectures 11 and 12. You should know that a nested class is a static class that is defined in another class, you should know how to draw nested classes (regular view or flattened view), and you should know the reasons for using a nested class. You may be asked to write a nested class. You should know that an inner class is a non-static class that is defined in another class, you should know how to draw inner classes (regular view or flattened view), and you should know the reasons for using an inner class. You may be asked to write an inner class.

**7. Algorithms.** We may ask you to write linear search, binary search, or the partition algorithm. These are presented on

pages 12-28 of the slides for lecture 06. We expect you to be able to give the specification of the problem (or to work with one we give you), write the invariant of the loop, and develop the loop as shown in lecture 06. With regard to binary search, you must know the algorithm as we give it in lecture 06.

**Here are sample questions (look also at previous prelims, on the web page, and attend the review session):**

1. Write a recursive method for computing a*b for integer a and non-negative integer b; you can't use multiplication.

2. Write a recursive method for setting each element of an array to its negation.

3. Write a recursive method that makes changes each character of a String s to lowercase.

4. Define "base case". Write the steps in understanding a recursive method.

4. Write the steps in evaluating a new expression.

5. Write the steps in executing the constructor call in the expression **new** C(a+b);

6. Define "inner class". Define "nested class".

7. We may give you a specification of a class and ask you to write the class.

8. Consider a class that implements interface CS211List using an array. Define the variables that will be used to contain the stack (and right comments that explain how the stack is stored in the array). Write a toString method that yields the a representation of the stack, with ", "
between adjacent elements.

9. We may give you a program that contains two classes and ask you to make one an inner class of the other.

10. We may ask you to write part of a class that has to throw an exception (e.g. like method getItem in class TagEnumeration).

11. We may give you a class and a few subclasses and ask you to make the class into an abstract class (and some of its methods into abstract methods) and to explain what the purpose of this is.

12. We may ask you to write a method that performs a linear search or binary search on any array of Comparable elements.