The final is cumulative, covering all topics in the course except as described below. So, you have to know everything that was covered in the two prelims in addition to material presented after those two prelims. See the handouts on the two prelims (on the course web page).

You do not have to study the following topics: reading a file or the keyboard, applications, applets, or interfaces (so ignore any questions on previous semesters’ exams about interfaces).

In addition to the material covered in the prelims, you do have to know about:

1. Several algorithms. You know this already, but we repeat it for emphasis. One of the following algorithms can be asked for. We may simply write “show binary search”, or “Show us the partition algorithm”, and you have to give the precondition, postcondition, and loop invariant and then develop the algorithm. Or, we may give you the header of the method and you have to write the precondition and postcondition that goes with it and then develop the rest. We expect that: the loop with initialization is developed from the invariant; a loop that has nothing to do with the invariant gets little credit. Everyone should get full credit on this question because it is simply a matter of (1) memorizing specifications and then (2) practicing developing known algorithms from their specs. For selection sort and insertion sort, write only a single loop, not a nested loop, as explained ad nauseam in lectures and in powerpoint slides, and the text.

Linear search, Binary search, Dutch National Flag, Partition algorithm, Selection sort, Insertion sort.

2. Developing an algorithm: stepwise refinement. We have used stepwise refinement in class many times, attempting to solve a little bit of a problem at a time. Read Sec. 2.5 on p. 82, and you might also study Sec. 9.2, p. 304, which discusses the development of several problems that deal with arrays.

3. Arrays. You have to know about one-dimensional, rectangular, and ragged arrays. This includes knowing how to access the number of columns in a row and knowing how to create a rectangular array or a ragged array. You have to know how arrays are stored as objects (folders) and to be able to draw an array.

4. Exception handling. Be able to write a class that extends Throwable, Exception, or RuntimeException, including the two constructors. Be able to write code to create an instance of such a class and throw it. Be able to write a simple try-statement with a single catch-clause. Understand what happens when an exception is thrown. Do not concern yourself with the throws clause; you won’t need it. See the chapter on Exception Handling.

5. Abstract classes. Know the purpose of making a class abstract and the syntax for it. Know the purpose of making a method abstract and the syntax for it. See Sec. 4.7 of the text and lesson page 4-5 of the ProgramLive CD.

6. Placement of components in a GUI. Know the default layout managers for a JFrame, a JPanel, and a Box and how that manager arranges components in it. Know these basic components: JButton, JLabel, JText...
Field, JTextArea. Know the three things you have to do to be able to listen to an event. Be able to understand programs that place components in a GUI and the code for listening to an event. You do not have to write code.