Review

I. Java
   A. Classes
      1. Write a Java class
      2. toString, equals, hashCode, implement Comparable interface
   B. Collections (Sets, Lists, Maps of all types)
   C. Deep Copy vs. Shallow Copy
      1. Explain what they are and how they are different
   D. Exceptions: declaring, throwing, and catching
   E. Inheritance
   F. Inner classes including anonymous inner classes
   G. Object Serialization
      1. Be able to explain what it is and why you would want to use it.

II. Java I/O
   1. Be familiar with the major I/O data types in Java, and be able to explain what they are (InputStream, BufferedInputStream, FileInputStream, Scanner, ObjectInputStream, etc.).
   2. Be able to write a piece of code that uses the Java I/O data types to perform a simple input task.

II. Databases
   A. Be able to design a set of database tables that store the data specified in a problem description (primary keys, foreign keys, know how to model one-to-one, one-to-many, and many-to-many relationships)
   B. Know basic SQL syntax for the following operations (your syntax need not be perfect, but should be pretty close)
      1. Create table statements
      2. Insert a row into a table
      3. Delete rows from a table
      4. Query: SELECT columns FROM tables WHERE condition
         a. Know how to use JOINS to perform a multi-table query

III. Designing and Writing High Quality Code:
A. Study online lecture notes for “software design” and “writing quality code”

B. Be conversant in the following concepts/principles: abstraction, cohesion/single Responsibility, abstracting all the way (i.e., avoid primitive obsession), minimize dependencies, separation of interface and implementation, information hiding, avoid code duplication, comments, good names, indenting, whitespace, conditions, curly braces, statement per line, parameters, deep nesting, wrapping long lines, pseudo-code

IV. XML/JSON
   A. Different types of XML/JSON parsers (tree parsers and token/stream parsers)

V. Unit testing
   A. What is it? How do you do it? Why is it important?
   B. How you did unit testing on your projects

VI. Layouts and Widgets (all kinds of widgets used in project)
   A. As they appear in the XML file (basic attributes, including IDs)
   B. Attaching listeners to widgets
   C. Displaying toasts
   D. Widgets used in project: Button, TextView, EditText, Spinner, Switch, RecyclerView, LinearLayout, RelativeLayout, GridLayout, etc.
   E. Understand RecyclerView architecture. What are View Holder for? What is the Adapter class for?

VII. Activities
   A. Be able to describe the Activity lifecycle
   B. Calling another activity
   C. Returning results from an activity to its calling activity
   D. Use of Intents and Bundles
      1. Adding Extras

VIII. The Android toolbar
   A. Creating and handling options menu

IX. Fragments
   A. What are fragments for? Why are they useful?
X. Web Access
   A. HTTP protocol basics
      1. URL format
      2. Contents/format of HTTP request and response messages
      3. Difference between GET and POST requests
      4. How to access a web resource in Java (HttpURLConnection)

XI. Defensive Programming
   A. Assertions
   B. Parameter checking
      1. Enforce with assertions or exceptions

XII. Software Testing
   A. Black Box Testing
      1. Equivalence Partitioning
      2. Boundary Value Analysis
   B. White Box Testing
      1. Line/Statement Coverage
      2. Branch Coverage
      3. Complete Condition Testing
      4. Loop Testing
      5. Relational Testing