1. A. Scroll sensitive is a result set type parameter, and updatable is a concurrency mode. The result set type parameter is passed to `createStatement()` before the concurrency mode. If you request options that the database driver does not support, it downgrades to an option it does support rather than throwing an exception. Statements I and III
are correct, making Option A the answer.

2. B. JDBC 4.0 allows, but does not require, a call to the `Class.forName()` method. However, since it is in the code, it needs to be correct. This method is expecting a fully qualified class name of a database driver, not the JDBC URL. As a result, the `Class.forName()` method throws a `ClassNotFoundException`, and Option B is the answer.

3. B. There are two `ResultSet` concurrency modes: `CONCUR_READ_ONLY` and `CONCUR_UPDATABLE`. All database drivers support read-only result sets, but not all support updatable ones. Therefore, Option B is correct.

4. D. This code is missing a call to `rs.next()`. As a result, `rs.getInt(1)` throws a `SQLException` with the message `Invalid cursor state - no current row`. Therefore, Option D is the answer.

5. D. The `execute()` method is allowed to run any type of SQL statements. The `executeUpdate()` method is allowed to run any type of the SQL statement that returns a row count rather than a `ResultSet`. Both `DELETE` AND `UPDATE` SQL statements are allowed to be run with either `execute()` or `executeUpdate()`. They are not allowed to be run with `executeQuery()` because they do not return a `ResultSet`. Therefore, Option D is the answer.

6. C. `Connection` is an interface rather than a concrete class. Therefore, it does not have a constructor and line `s2` does not compile. As a result, Option C is the answer. Option A would be the answer if the code `new Connection()` was changed to `DriverManager.getConnection()`.

7. A. There are three `ResultSet` type options: `TYPE_FORWARD_ONLY`, `TYPE_SCROLL_INSENSITIVE`, and `TYPE_SCROLL_SENSITIVE`. Only one of these is in the list, making Option A correct.

8. B. Unlike arrays, JDBC uses one-based indexes. Since `num_pages` is in the second column, the parameter needs to be `2`, ruling out Options A and C. Further, there is not a method named `getInteger()` on the `ResultSet` interface, ruling out Option D. Since the proper method is `getInt()`, Option B is the answer.

9. D. Option A does not compile because you have to pass a column index or column name to the method. Options B and C compile. However, there are not columns named `0` or `1`. Since these column names don't exist, the code would throw a `SQLException` at runtime. Option D is correct as it uses the proper column name.

10. B. The parameters to `createStatement()` are backward. However, they still compile because both are of type `int`. This means the code to create the `Statement` does compile, and Option A is incorrect. Next comes the code to create the `ResultSet`. While both `execute()` and `executeQuery()` can run a `SELECT` SQL statement, they have different return types. Only `executeQuery()` can be used in this example. The code does not compile because the `execute()` method returns a `boolean`, and Option B is correct. If this was fixed, Option D would be the answer because `rs.next()` is never called.

11. D. Since this code opens `Statement` using a try-with-resources, `Statement` gets closed automatically at the end of the block. Further, closing a `Statement` automatically closes a `ResultSet` created by it, making Option D the answer. Remember that you should close any resources you open in code you write.

12. C. Option A is incorrect because `Driver` is an interface while `DriverManager` is a concrete class. The inverse isn't true either; `DriverManager` doesn't implement `Driver`. Option B is incorrect because the `Connection` implementation comes from the database driver jar. Option C is correct. You can turn off auto-commit mode, but it defaults to on. Option D is incorrect because you need to call `rs.next()` or an equivalent method to point to the first row.

13. C. The requirement to include a `java.sql.Driver` file in the `META-INF` directory was introduced in JDBC 4.0. Older drivers are not required to provide it, making Option B incorrect. A file named `jdbc.driver` has never been a requirement. Option A is incorrect and is simply here to trick you. All drivers are required to implement the `Connection` interface, making Option C the answer.

14. D. First, `rs.next()` moves the cursor to point to the first row, which contains the number `10`. Line `q1` moves the cursor to immediately before the first row. This is the same as the position it was in before calling`rs.next()` in the first place. It is a valid position but isn't a row of data. Line `q2` tries to retrieve the data at this position and throws a `SQLException` because there isn't any data, making Option D the answer.

15. B. This code shows how to properly update a `ResultSet`. Note that it calls `updateRow()` so the changes get applied in the database. This allows the `SELECT` query to see the changes and output `10`. Option B is correct. Remember that unlike this code, you should always close a `ResultSet` when you open it in real code.