Chapter 17 Exceptions and Assertions

1. See the section "Exceptions and Exception Types." The Throwable class is the root of Java exception classes. Error and Exception are subclasses of Throwable. Error describes fatal system errors, and Exception describes the errors that can be handled by Java programs. The subclasses of Error are LinkageError, VirtualMachineError, and AWTError. The subclasses of Exception include RuntimeException, IOException, AWTException, and InstantiationException.

   a. ArithmeticException
   b. ArrayIndexOutOfBoundsException
   c. StringIndexOutOfBoundsException
   d. ClassCastException
   e. NullPointerException
   f. No exception

2. 
   a. 0 1
   b. 0

3. The purpose of claiming exceptions is to tell the Java runtime system what can go wrong. You claim an exception using the throws keyword in the method declaration. You can claim multiple exceptions, separated by commas.

4. A checked exception must be explicitly declared in the method declaration, if a method throws it. A checked exception must be caught in a try-catch block. An unchecked exception does not need to be declared and does not need to be caught. In Java, the only unchecked exceptions are RuntimeException and Error and their subclasses.

5. You use the throw statement in the method to throw an exception. You cannot throw multiple exceptions in a single throw statement.

6. throw is for throwing exceptions and throws is for claiming exceptions.

7. When an exception occurs, the Java runtime system creates an object for the exception, and the catch process starts. To catch exceptions, use a try-catch block to catch exceptions.

8. The printout is

   value is too small
   Continue after the catch block

   The printout would be if Line int value = 30; is changed to int value = 50;
9. Will statement3 be executed?
   **Answer:** No.

   If the exception is not caught, will statement4 be executed?
   **Answer:** No.

   If the exception is caught in the catch clause, will statement4 be executed?
   **Answer:** Yes.

   If the exception is passed to the caller, will statement4 be executed?
   **Answer:** No.

10. RuntimeException
    Reason: list[10] throws ArrayIndexOutOfBoundsException that is a subclass of RuntimeException.

11. ArithmeticException
    Reason: method() throws ArithmeticException.

12. RuntimeException in method()
    After the method call
    Reason: s.charAt(3) throws StringIndexOutOfBoundsException that is a subclass of RuntimeException. This exception is caught in method(). The main method continues its normal execution flow.

13. If an exception were not caught in a non-GUI application, the program would terminate. If an exception were not caught in a GUI application, the program would continue and the error message would be reported on the console.

14. To print trace information to the console.

15. No.

16. The method throws a checked exception. It must be caught or thrown. You may fix it as follows:

   ```java
   public void m(int value) throws Exception {
       if (value < 40)
           throw new Exception("value is too small");
   }
   ```

17. Will statement5 be executed if the exception is not caught?
   **Answer:** No.
If the exception is of type Exception3, will statement 4 be executed? Will statement 5 be executed?

**Answer:** This exception is caught by the `catch (Exception3 e3)` clause and statement 4 will be executed, but statement 5 will not be executed because it is rethrown to its caller.

18. Exception in method()
    Exception in main
    Reason: the setRadius method throws a RadiusException. RadiusException is a subclass of Exception. So it is caught in method()’s handler. The handler rethrows it back to the main method.

19. An assertion is a Java statement that enables you to assert an assumption about your program. An assertion contains a Boolean expression that should be true during program execution. To declare an assertion, use the assert key word followed by a Boolean expression with an optional value. To compile the code with assertion, use the –source 1.4 switch in the javac command. To run the code with assertion, use the –ea switch in the java command.

20. An java.lang.AssertionError would be thrown since the assertion is violated at Line 7 (i is actually 11).