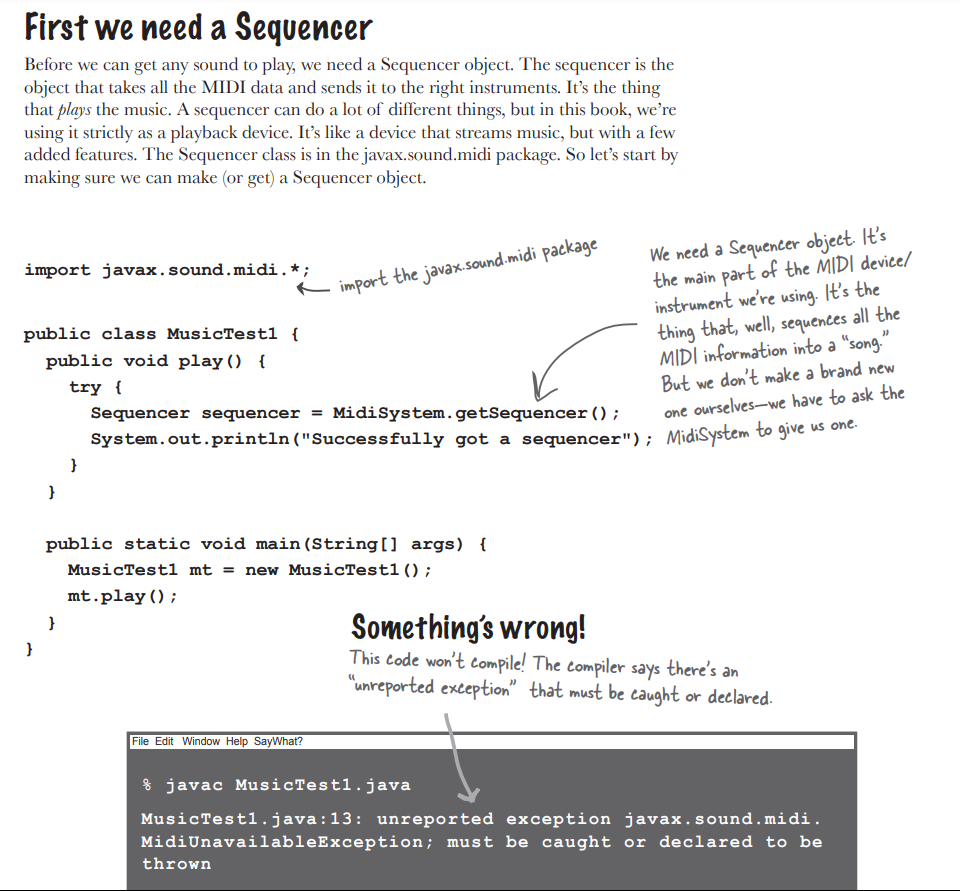
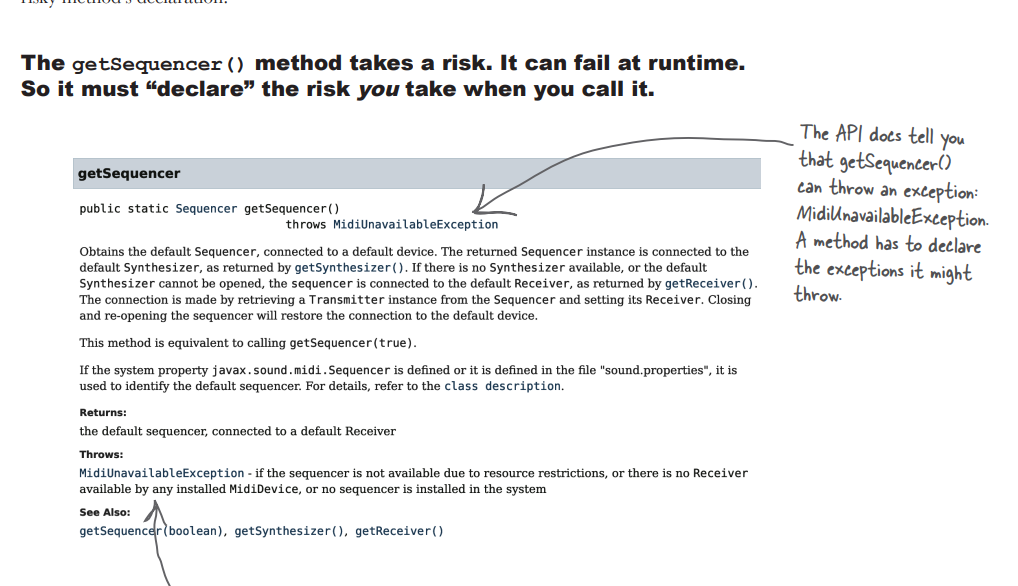
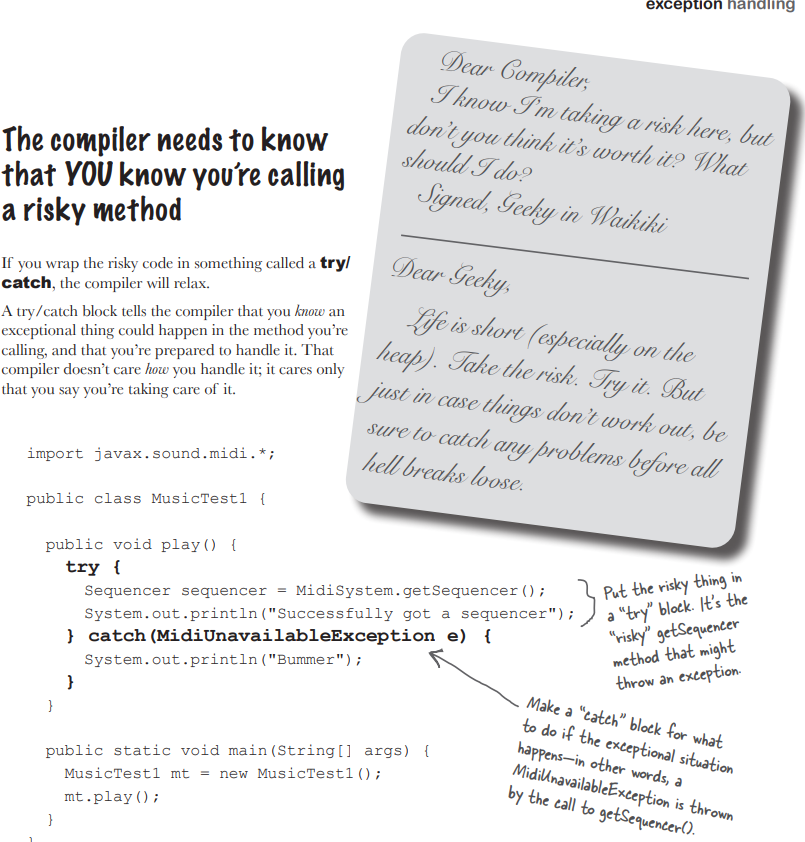
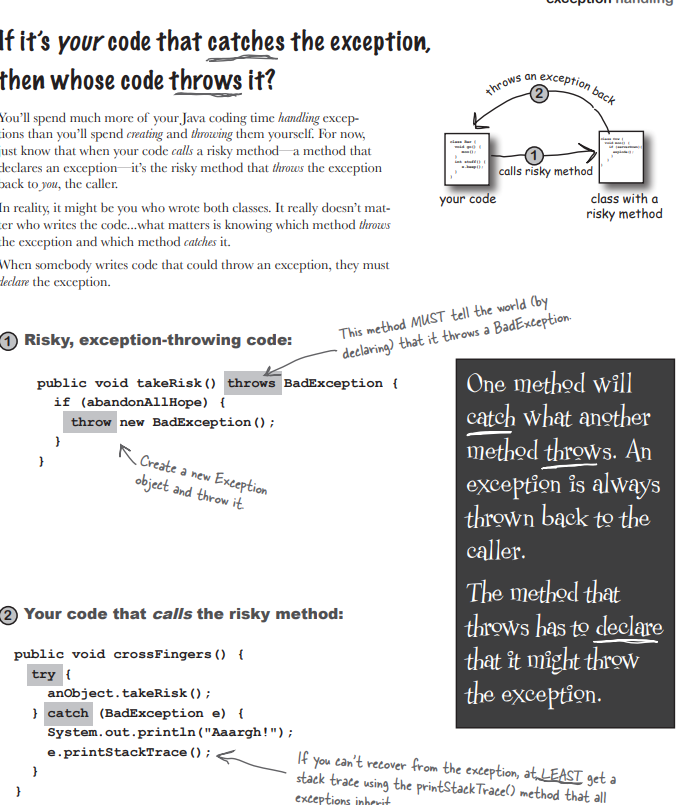
**CHAPTER – 13**

**Risky Behavior**

****

****

****

****

****

****

**EXERCISE**

**1.TRUE OR FALSE**

1. A try block must be followed by a catch and a finally block. - False
2. If you write a method that might cause a compiler-checked exception, you must wrap that risky code in a try/catch block. - False
3. Catch blocks can be polymorphic. - true
4. Only “compiler checked” exceptions can be caught. - False
5. If you define a try/catch block, a matching finally block is optional.- True
6. If you define a try block, you can pair it with a matching catch or finally block, or both.- True
7. If you write a method that declares that it can throw a compiler-checked exception, you must also wrap the exception throwing code in a try/catch block.- False
8. The main() method in your program must handle all unhandled exceptions thrown to it.- False
9. A single try block can have many different catch blocks.- True
10. A method can throw only one kind of exception.-True
11. A finally block will run regardless of whether an exception is thrown.-True
12. A finally block can exist without a try block.- False
13. A try block can exist by itself, without a catch block or a finally block.- False
14. Handling an exception is sometimes referred to as “ducking.” – False
15. The order of catch blocks never matters.- False
16. A method with a try block and a finally block can optionally declare a checked exception.- False
17. Runtime exceptions must be handled or declared.- False

**2.CODE MAGNETS**

class MyEx extends Exception { }

public class ExTestDrive {

public static void main(String[] args) {

String test = args[0];

try {

System.out.print("t");

doRisky(test);

System.out.print("o");

} catch (MyEx e) {

System.out.print("a");

} finally {

System.out.print("w");

}

System.out.println("s");

}

static void doRisky(String t) throws MyEx {

System.out.print("h");

if ("yes".equals(t)) {

throw new MyEx();

}

System.out.print("r");

}

}