Chapter 14

Testing and debugging

Lesson page 14-1. Introduction to testing and debugging

Activity 14-1-1 Good programming practices

Question 1. A bug is an unexpected defect or flaw, as in a plan, mechanism, or piece of legislation. In programming, a bug is a mistake in a program.

Question 2. To debug is to detect and eliminate the mistakes or malfunctions in. In programming, to debug means to find an eliminate the errors in a program.

Question 3. A test case is one set of inputs to a program (or method, or class) that is used to test whether the program (or method, or class) meets its specifications

Question 4. The underlined terms are: specification; statement-comments; meaning; documentation; specification; implementation; definition; statements.

Lesson page 14-2. Testing strategies

Activity 14-2-1 Using GUI `JLiveWindow` as a test driver

Question 1. A test driver is a program whose sole purpose it to aid in testing a method or class (or other program segment).

Question 2. You have to change method `main`, in order to have the correct fields displayed in the GUI; and you have to change method `buttonPressed`, to get values from the GUI, exercise the program segment to be tested, and display the results in the GUI or Java console.

Activity 14-2-2 Another test driver for a method

Question 3. `MyJLiveWindow` might be preferred, because it can test many inputs in one run of the program. The Java console might be preferred, because it is easier to set up.

Question 4. Here's the loop:

```
while (true) {
    System.out.println("Type an argument");
    double b= JLiveRead.readLineDouble();
    System.out.println(TempConvert.KelvFromFahr(b));
}
```

Activity 14-2-3 The test driver for a class

Question 5. The test driver for a class should (1) declare variables of the class-type; (2) store new instances of the class in the variables; (3) exercise and test the constructors and method `toString`; and (4) exercise and test all other methods of the class.

Question 6. Method `toString` will be helpful in testing almost all properly designed classes.

Activity 14-2-4 Assertions in testing

Question 7. The underlined word is: assertions.

Lesson page 14-3. Selecting test cases and checking them

Activity 14-3-1 Structural testing

Question 1. Draw a line from blackbox testing to functional testing and from whitebox testing to structural testing.

Question 2. The guidelines for structural testing are: (1) test each method (or group of methods) thoroughly as it is completed; (2) try some simple test cases; (3) provide test coverage —so that each program part is executed at least once; (4) provide boundary cases; and (5) include illegitimate cases.

Activity 14-3-2 Checking test cases automatically

Question 3. In automatic testing, there will be output for a test case only if the test case detects an error. Therefore, if there is no output —there is silence from the program— the program passes all test cases.

Activity 14-3-3 Partial checking done automatically

Question 4. Two ideas for partial testing are: (1) write a program segment to check the result of some test cases, and (2) use a previously written method that allows for some partial tests.

Lesson page 14-4. Debugging

Activity 14-4-1 Tracking down a bug

Question 1. The underlined words are: beginning; end

Activity 14-4-2 Tracking down another bug

Question 2. The two guidelines are: (1) glean as much information as possible from each debugging run and (2) insert print statements at judiciously chosen places, in order to check the values of variables.