Common Errors in Method Exercises

For Introduction to Java Programming by Y. Daniel Liang

A common error for the exercises in this chapter on methods is that students don't implement the methods to meet the specification requirements even though the output from the main program is correct.

Here are two examples:

Example 1: Suppose you are asked to write method that returns a pentagonal number. A pentagonal number is defined as n(3n–1)/2 for $n = 1, 2, \ldots$, and so on. Therefore, the first few numbers are 1, 5, 12, 22, ... Write a method with the following header that returns a pentagonal number:

```
public static int getPentagonalNumber(int n)
```

Write a test program that uses this method to display the first 100 pentagonal numbers with 10 numbers on each line.

Two programs are given to solve this problem in Listing A and Listing B. Both programs produce the same output, but the program in Listing A is wrong, because the implementation of the method does not meet the requirements. The requirement for the method is simply to return a pentagonal number (not to print all 100 pentagonal numbers.)

Listing A IncorrectPentagonalNumberMethodImplementation.java

```
public class IncorrectPentagonalNumberMethodImplementation {
  public static void main(String[] args) {
    System.out.println("Pentagonal Numbers: ");
    getPentagonalNumber(100);
  }

  public static int getPentagonalNumber(int n) {
    int number = 1;

    while (number <= 100) {
      int pentagonalNumber = number * (3 * number - 1) / 2;
      if (number % 10 == 0)
        System.out.println(pentagonalNumber);
      else
        System.out.print(pentagonalNumber + " ");

      number++;
    }

    return number;
  }
}
```

Listing B CorrectPentagonalNumberMethodImplementation.java

```
public class CorrectPentagonalNumberMethodImplementation {
  public static void main(String[] args) {
    System.out.println("The first 100 pentagonal numbers, ten per line: ");
    for (int n = 1; n <= 100; n++) {
      if (n % 10 == 0)
        System.out.printf("%7d\n", getPentagonalNumber(n));
      else
        System.out.printf("%7d", getPentagonalNumber(n));
    }
  }

  /** Returns the pentagonal number for n */
```

```
  public static int getPentagonalNumber(int n) {
    return n * (3 * n - 1) / 2;
  }
}
```

Example 2: Suppose you are asked to write the following method that returns the sum of the following series:

$$m(n) = \frac{3}{1} + \frac{5}{4} + \frac{7}{7} + \frac{9}{10} + \frac{11}{13} + \frac{13}{16} + ... + \frac{2n+1}{3n-2}$$

```
public static double m(int n)
```

Write a test program that uses this method to display a table like this:

```
             n           m(n)
----------------------------------------
             1           3.00
             2           4.25
             3           5.25
             4           6.15
             5           7.00
             6           7.81
             7           8.60
             8           9.37
             9          10.13
            10          10.88
```

Two programs are given to solve this problem in Listing 1 and Listing 2. Both programs produce the same output, but the program in Listing 1 is wrong, because the implementation of the method m does not meet the requirements. The requirement is to return the sum of all items, but your implementation returns only the last item.

Listing 1 IncorrectMethodImplementation.java

```
public class IncorrectMethodImplementation {
  public static void main(String[] args) {
    System.out.printf("%15s%15s\n", "n", "m(n)");
    System.out.println("----------------------------------------");
    for (int n = 1; n <= 10; n++) {
      double result = 0;
      for (int i = 1; i <= n; i++) {
        result += m(i);
      }
      System.out.printf("%15d%15.2f\n", n, result);
    }
  }

  public static double m(int n) {
    return (2 * n + 1.0) / (3 * n - 2);
  }
}
```

Listing 2 CorrectMethodImplementation.java

```
public class CorrectMethodImplementation {
  public static void main(String[] args) {
    System.out.printf("%15s%15s\n", "n", "m(n)");
    System.out.println("----------------------------------------");
    for (int i = 1; i <= 10; i++) {
```

```
      System.out.printf("%15d%15.2f\n", i, m(i));
    }
  }

  public static double m(int n) {
    double result = 0;
    for (int i = 1; i <= n; i++) {
      result += (2 * i + 1.0) / (3 * i - 2);
    }
    return result;
  }
}
```