Building Java Programs

Chapter 5
Lecture 5-4: Assertions

reading: 5.5
HUMANS HAVEN'T PROGRAMMED ANYTHING IN DECADES. ALL THE LANGUAGES AND IDEAS AND JARGON ARE JUST TOYS IN THE ROBOTS' SANDBOX. THE REAL PROGRAMMING HAPPENS AT A LOWER LEVEL, BUT NONE OF THE PROGRAMMERS KNOW IT.

WEIRD... THIS SUBROUTINE WORKS NOW, BUT I SWEAR I DIDN'T CHANGE A THING.

NOWADAYS, WE'RE JUST PART OF THE JUNK CODE. DON'T BELIEVE ME? GO AHEAD—COMPARO PROGRAMMER SPEAK TO GIBBERISH-GENERATING SPAMBOTS. CAN YOU TELL THE DIFFERENCE?

I USE PYLIBMC TO TALK TO MEMCACHED FROM DJANGO.

AFFT? RUBY, MAN? RUBY!

Logical assertions

- **assertion**: A statement that is either true or false.

  Examples:
  - Java was created in 1995.
  - The sky is purple.
  - 23 is a prime number.
  - 10 is greater than 20.
  - x divided by 2 equals 7. *(depends on the value of x)*

- An assertion might be false ("The sky is purple" above), but it is still an assertion because it is a true/false statement.
Reasoning about assertions

• Suppose you have the following code:

```java
if (x > 3) {
    // Point A
    x--;  
} else {
    // Point B
    x++;  
    // Point C
}
// Point D
```

• What do you know about x's value at the three points?
  • Is x > 3? Always? Sometimes? Never?
Assertions in code

- We can make assertions about our code and ask whether they are true at various points in the code.
- Valid answers are ALWAYS, NEVER, or SOMETIMES.

```java
System.out.print("Type a nonnegative number: ");
double number = console.nextDouble();
// Point A: is number < 0.0 here? (SOMETIMES)

while (number < 0.0) {
    // Point B: is number < 0.0 here? (ALWAYS)
    System.out.print("Negative; try again: ");

    number = console.nextDouble();
    // Point C: is number < 0.0 here? (SOMETIMES)
}

// Point D: is number < 0.0 here? (NEVER)
```
Reasoning about assertions

- Right after a variable is initialized, its value is known:
  ```java
  int x = 3;
  // is x > 0? ALWAYS
  ```

- In general you know nothing about parameters' values:
  ```java
  public static void mystery(int a, int b) {
    // is a == 10? SOMETIMES
  }
  ```

- But inside an `if`, `while`, etc., you may know something:
  ```java
  public static void mystery(int a, int b) {
    if (a < 0) {
      // is a == 10? NEVER
      ...
    }
  }
  ```
Assertions and loops

- At the start of a loop's body, the loop's test must be **true**:
  
  ```java
  while (y < 10) {
      // is y < 10?  ALWAYS
      ...
  }
  ```

- After a loop, the loop's test must be **false**:
  
  ```java
  while (y < 10) {
      ...
  }
  // is y < 10?  NEVER
  ```

- Inside a loop's body, the loop's test may become **false**:
  
  ```java
  while (y < 10) {
      y++;
      // is y < 10?  SOMETIMES
  }
  ```
"Sometimes"

- Things that cause a variable's value to be unknown (often leads to "sometimes" answers):
  - reading from a `Scanner`
  - reading a number from a `Random` object
  - a parameter's initial value to a method

- If you can reach a part of the program both with the answer being "yes" and the answer being "no", then the correct answer is "sometimes".
  - If you're unsure, "Sometimes" is a good guess.
Assertion example 1

```java
public static void mystery(int x, int y) {
    int z = 0;

    // Point A
    while (x >= y) {
        // Point B
        x = x - y;
        z++;
        if (x != y) {
            // Point C
            z = z * 2;
        }
        // Point D
    }

    // Point E
    System.out.println(z);
}
```

Which of the following assertions are true at which point(s) in the code? Choose ALWAYS, NEVER, or SOMETIMES.

<table>
<thead>
<tr>
<th></th>
<th>x &lt; y</th>
<th>x == y</th>
<th>z == 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point A</td>
<td>SOMETIMES</td>
<td>SOMETIMES</td>
<td>ALWAYS</td>
</tr>
<tr>
<td>Point B</td>
<td>NEVER</td>
<td>SOMETIMES</td>
<td>SOMETIMES</td>
</tr>
<tr>
<td>Point C</td>
<td>SOMETIMES</td>
<td>NEVER</td>
<td>NEVER</td>
</tr>
<tr>
<td>Point D</td>
<td>SOMETIMES</td>
<td>SOMETIMES</td>
<td>NEVER</td>
</tr>
<tr>
<td>Point E</td>
<td>ALWAYS</td>
<td>NEVER</td>
<td>SOMETIMES</td>
</tr>
</tbody>
</table>
public static int mystery(Scanner console) {
    int prev = 0;
    int count = 0;
    int next = console.nextInt();

    // Point A
    while (next != 0) {
        // Point B
        if (next == prev) {
            // Point C
            count++;
        }
        prev = next;
        next = console.nextInt();
        // Point D
    }
    // Point E
    return count;
}

Which of the following assertions are true at which point(s) in the code? Choose ALWAYS, NEVER, or SOMETIMES.

<table>
<thead>
<tr>
<th></th>
<th>next == 0</th>
<th>prev == 0</th>
<th>next == prev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point A</td>
<td>SOMETIMES</td>
<td>ALWAYS</td>
<td>SOMETIMES</td>
</tr>
<tr>
<td>Point B</td>
<td>NEVER</td>
<td>SOMETIMES</td>
<td>SOMETIMES</td>
</tr>
<tr>
<td>Point C</td>
<td>NEVER</td>
<td>NEVER</td>
<td>ALWAYS</td>
</tr>
<tr>
<td>Point D</td>
<td>SOMETIMES</td>
<td>NEVER</td>
<td>SOMETIMES</td>
</tr>
<tr>
<td>Point E</td>
<td>ALWAYS</td>
<td>SOMETIMES</td>
<td>SOMETIMES</td>
</tr>
</tbody>
</table>
// Assumes y >= 0, and returns x^y
public static int pow(int x, int y) {
    int prod = 1;

    // Point A
    while (y > 0) {
        // Point B
        if (y % 2 == 0) {
            // Point C
            x = x * x;
            y = y / 2;
            // Point D
        } else {
            // Point E
            prod = prod * x;
            y--;
            // Point F
        }
    }

    // Point G
    return prod;
}

Which of the following assertions are true at which point(s) in the code?
Choose ALWAYS, NEVER, or SOMETIMES.

<table>
<thead>
<tr>
<th>Point</th>
<th>y &gt; 0</th>
<th>y % 2 == 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point A</td>
<td>SOMETHING</td>
<td>SOMETHING</td>
</tr>
<tr>
<td>Point B</td>
<td>ALWAYS</td>
<td>SOMETHING</td>
</tr>
<tr>
<td>Point C</td>
<td>ALWAYS</td>
<td>ALWAYS</td>
</tr>
<tr>
<td>Point D</td>
<td>ALWAYS</td>
<td>SOMETHING</td>
</tr>
<tr>
<td>Point E</td>
<td>ALWAYS</td>
<td>NEVER</td>
</tr>
<tr>
<td>Point F</td>
<td>SOMETHING</td>
<td>ALWAYS</td>
</tr>
<tr>
<td>Point G</td>
<td>NEVER</td>
<td>ALWAYS</td>
</tr>
</tbody>
</table>