Brendan Whitaker

CSE 2221 Homework 12

Professor Bucci

7 March 2017

1. We implement the given static method:

```
/**
 * Returns the number of digits of {@code n}.
 *
 * @param n
 *            {@code NaturalNumber} whose digits to count
 * @return the number of digits of {@code n}
 * @ensures numberOfDigits = [number of digits of n]
 */
private static int numberOfDigits(NaturalNumber n) {
    int numDigits = 0;
    if (!n.isZero()) {
        n.divideBy10();

        numDigits += numberOfDigits(n);
        numDigits++;
    }

    return numDigits;
}
```

2. We implement the given static method:

```
/**
 * Returns the sum of the digits of {@code n}.
 *
 * @param n
 *            {@code NaturalNumber} whose digits to add
 * @return the sum of the digits of {@code n}
```

```
 * @ensures sumOfDigits = [sum of the digits of n]
 */
private static int sumOfDigits(NaturalNumber n) {
    int sumDigits = 0;
    if (!n.isZero()) {

        sumDigits = n.divideBy10();
        sumDigits += sumOfDigits(n);
    }
    return sumDigits;
}
```

3. We implement the given static method:

```
/**
 * Returns the sum of the digits of {@code n}.
 *
 * @param n
 *            {@code NaturalNumber} whose digits to add
 * @return the sum of the digits of {@code n}
 * @ensures sumOfDigits = [sum of the digits of n]
 */
private static NaturalNumber sumOfDigits(NaturalNumber n) {
    NaturalNumber digits1 = n.newInstance();
    if (!n.isZero()) {
        NaturalNumber digits = new NaturalNumber2(n.divideBy10());
        digits1.transferFrom(digits);
        digits1.add(sumOfDigits(n));

    }
    return digits1;
}
```

4. We implement the given static method:

```
/**
 * Divides {@code n} by 2.
 *
 * @param n
 *            {@code NaturalNumber} to be divided
 * @updates n
 * @ensures 2 * n <= #n < 2 * (n + 1)
 */
private static void divideBy2(NaturalNumber n) {
```

```
    if (!n.isZero()) {
        int digit = n.divideBy10();
        digit /= 2;
        int digit2 = n.divideBy10();
        if (digit2 % 2 == 1) {
            digit += 5;
            if (digit > 9) {
                n.multiplyBy10(digit2 + 1);
                digit -= 10;
            } else {
                n.multiplyBy10(digit2);
            }
        }
        divideBy2(n);
        n.multiplyBy10(digit);
    }
}
```

5. We implement the given static method:

```
/**
 * Checks whether a {@code String} is a palindrome.
 *
 * @param s
 *            {@code String} to be checked
 * @return true if {@code s} is a palindrome, false otherwise
 * @ensures isPalindrome = (s = rev(s))
 */
private static boolean isPalindrome(String s) {
    boolean pal = true;
    int length = s.length();
    if (length > 0) {
        if (s.charAt(0) == s.charAt(length - 1)) {
            if (length > 2 && !(isPalindrome(s.substring
            (1, length - 1)))) {
                pal = false;
            }
        } else {
            pal = false;
        }
    }
    return pal;
}
```