```java
import components.naturalnumber.NaturalNumber;
import components.naturalnumber.NaturalNumber2;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;

/**
 * Put a short phrase describing the program here.
 *
 * @author Sam Espanioly
 *
 */
public final class HW12 {

    /**
     * Private constructor so this utility class cannot be instantiated.
     */
    private HW12() {
    }

    //Implement the static method declared as follows:
    /**
     * 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 digit = 0;
        int rem = n.divideBy10();
        digit++;
        if (!n.isZero()) {
            // adds it after every recursion
            digit = digit + numberOfDigits(n);

        }
        n.multiplyBy10(rem);
        return digit;
    }

    //Implement the static method declared as follows:
    /**
     * 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 remainder = n.divideBy10();

        int sum = remainder;
        if (!n.isZero()) {
            //adds it after it returns it after calling it
            sum = sum + sumOfDigits(n);
        }
        n.multiplyBy10(remainder);
        return sum;
    }

    //In addition to the kernel methods, for this question
```

```
(only!) you are allowed to use the NaturalNumber method
add. Implement the static method declared as follows:
    /**
     * 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
sumOfDigits2(NaturalNumber n) {
        int temp = n.divideBy10();

        NaturalNumber sum = new NaturalNumber2(temp);
        if (!n.isZero()) {
            sum.add(sumOfDigits2(n));
            //sum = sum + sumOfDigits(n);
        }
        n.multiplyBy10(temp);
        return sum;
    }

    //Implement the static method declared as follows:
    /**
     * 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) {
        int rem = n.divideBy10();
        if (!n.isZero()) {
            // get next remainder
            int nextRem = n.divideBy10();
            //check if its even
            if (nextRem % 2 == 0) {
                rem = rem / 2;
            } else {//else its odd
                // add 5 in case next decimal is odd
                rem = rem / 2 + 5;
            }
            // added this so it divides the last digit
without trouble
            if (n.isZero()) {
                nextRem /= 2;
            }
            n.multiplyBy10(nextRem);
            divideBy2(n);

        }

        n.multiplyBy10(rem);
    }

    //Implement the static method declared as follows:
    /**
     * 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))
     */
```

```java
    private static boolean isPalindrome(String s) {
        boolean check = true;
        int i = 0;//first char
        int len = s.length();
        int u = len - 1;// last char
        // len / 2 + 1 to make it more efficient
        while (i < ((len / 2) + 1)) {
            //to avoid spaces
            if (s.charAt(i) == ' ') {
                i++;
            }
            //to avoid spaces
            if (s.charAt(u) == ' ') {
                u--;
            }
            // compare them
            if (s.charAt(i) != s.charAt(u)) {
                check = false;
                i = len;// added this to make the loop
quit fast== more efficient
            }
            u--;
            i++;
        }
        return check;
    }

    /**
     * Main method.
     *
     * @param args
     *            the command line arguments
     */
    public static void main(String[] args) {
        SimpleReader in = new SimpleReader1L();
        SimpleWriter out = new SimpleWriter1L();
        //testing methods
        out.println("Enter n: ");
        NaturalNumber n = new NaturalNumber2(in.nextLine());
        out.println(numberOfDigits(n));
        out.println(n);// check for restore n
        out.println(sumOfDigits(n));
        out.println(n);// check for restore n
        out.println(sumOfDigits2(n));
        out.println(n);// check for restore n
        divideBy2(n);
        out.println("divide by 2: " + n);
        //testing methods
        String s = "racecar";
        out.println(isPalindrome(s));
        s = "r ace c ar";
        out.println(isPalindrome(s));
        s = "not a racecar";
        out.println(isPalindrome(s));
        in.close();
        out.close();
    }
}

```