Solution 31-10-23

231. Power of Two

```
import java.util.Scanner;

public class PowerOfTwoChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        long number = scanner.nextLong();
        scanner.close();

        if (isPowerOfTwo(number)) {
            System.out.println(number + " is a power of two.");
        } else {
            System.out.println(number + " is not a power of two.");
        }
    }

    public static boolean isPowerOfTwo(long number) {
        if (number <= 0) {
            return false;
        }
        // A power of two has only one bit set in its binary representation.
        // If we subtract 1 from a power of two, all the bits to the right of the set bit become 1.
        // So, a power of two AND (power of two - 1) should equal 0.
        return (number & (number - 1)) == 0;
    }
}
```

Output

java -cp /tmp/gHNoXjiOw7 PowerOfTwoChecker

Enter a number: 4569

4569 is not a power of two.

233. Number of Digit

```
import java.util.Scanner;

public class NumberOfDigitOne {
    public static int countDigitOne(int n) {
        int count = 0;
        for (long i = 1; i <= n; i *= 10) {
            long divider = i * 10;
            count += (n / divider) * i + Math.min(Math.max((n % divider - i + 1), 0), i);
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();
        scanner.close();

        int numberOfOnes = countDigitOne(number);
        System.out.println("The number of times the digit '1' appears from 1 to " + number + "
is: " + numberOfOnes);
    }
```

}

Output

java -cp /tmp/2vSy9vkGR4 NumberOfDigitOne

Enter a number: 58

The number of times the digit '1' appears from 1 to 58 is: 16