Silly Counting

Johnny's daughter Cindy really likes numbers and counting. She made up the following counting game. You start a certain number and count up. Normally you just say the number, but in some cases you say something else:

- The number `7` looks like an upside down nose. When the counted number contains a `7` as a digit or can be divided by `7` you say `NOSE` instead of the number.
- The number `11` looks like two bars in a prison window. When the counted number contains `11` somewhere in its digits or can be divided by `11` you say `BARS` instead of the number.
- The number `89` looks like someone who is really enjoying dinner. When the counted number contains `89` or can be divided by `89` you say `YUMMY` instead of the number.
- If multiple of these rules are valid for a certain counted number, you say all relevant words in alphabetic order.

Create a class `SillyCounting` with the following public interface:

```
1 public class SillyCounting
2 {
3    public SillyCounting(int start) { ... }
4    public String getNextCount() { ... }
5 }
```

Make sure that the words are presented in upper-case letters. If multiple words occur they should be seperated by a single space. There should not be any leading or trailing spaces in the output.

The first time `getNextCount()` is called, it should say something based on the initial value that was passed to the constructor.

Example: Consinder the following calls to a `SillyCounting` object:

```
1 SillyCounting sc = new SillyCounting(6);
2 for (int t=0; t < 6; t++)
3 {
4   System.out.println(sc.getNextCount());
5 }
```

This should say `6`, `NOSE`, `8`, `9`, `10`, `BARS` on different lines without the comma's.

Aside from producing the correct output, your code should adhere to the following rules:

- All your instance variables need to be declared `private`.
- Every `public` method or constructor is required to have a Javadoc style comment.
- The Javadoc comment associated with the class itself must define an `@author` of the file.

Hints: For checking whether a number can be divided by another number you should use the modulo operator. To see whether a certain number appears among the digits of a number, you can make clever use of methods of the `String` class. There is no need to use clever mathematical tricks.