University of Washington
Computer Science & Engineering 142 (Computer Programming I), Summer 2005
Programming Assignment #1
Due: Tuesday, 6/28/2005, 6:00 PM

Program Description:

This program tests your understanding of using static methods and println statements. You should write a Java class called `Song` that should be saved into a file called `Song.java`. Your Song program should produce the following song as output:

```
There was an old woman who swallowed a fly.
I don't know why she swallowed that fly,
Perhaps she'll die.

There was an old woman who swallowed a spider,
That wriggled and iggled and jiggled inside her.
She swallowed the spider to catch the fly,
I don't know why she swallowed that fly,
Perhaps she'll die.

There was an old woman who swallowed a bird,
How absurd to swallow a bird.
She swallowed the bird to catch the spider,
She swallowed the spider to catch the fly,
I don't know why she swallowed that fly,
Perhaps she'll die.

There was an old woman who swallowed a cat,
Imagine that to swallow a cat.
She swallowed the cat to catch the bird,
She swallowed the bird to catch the spider,
She swallowed the spider to catch the fly,
I don't know why she swallowed that fly,
Perhaps she'll die.

There was an old woman who swallowed a dog,
What a hog to swallow a dog.
She swallowed the dog to catch the cat,
She swallowed the cat to catch the bird,
She swallowed the bird to catch the spider,
She swallowed the spider to catch the fly,
I don't know why she swallowed that fly,
Perhaps she'll die.

There was an old woman who swallowed a horse,
She died of course.
```

You should exactly reproduce the format of this output. This includes having identical wording, spelling, spacing, punctuation, and capitalization.

One way to write this program would be to simply write a println statement that outputs each line of the song in order. However, such a solution would not receive full credit. Part of the challenge of this assignment lies in recognizing the structure and redundancy of the song and improving the code using static methods.

Implementation Guidelines:

Use static methods to avoid simple redundancy in the output. In particular, you are to make sure that you use only one println statement for each distinct line of the song. For example, this line:

```
Perhaps she'll die.
```

appears several times in the output, but you should have only one println statement in your program that prints that line of the song.

There is more complex redundancy in the song that has to do with parts of pairs of lines, like these:

```
There was an old woman who swallowed a horse,
There was an old woman who swallowed a dog,
```

and like these (redundant parts appear in bold):

```
She swallowed the dog to catch the cat,
She swallowed the cat to catch the bird,
```

It is not possible to avoid this partial-line redundancy using just what we have learned so far (static methods and simple println statements), so you are not expected to eliminate it. There is, however, a general structural redundancy to the song that you can eliminate with static methods. The key question to ask yourself is whether or not you have repeated lines of code that could be eliminated if you structured your static methods differently.

You should also be using static methods to capture the structure of the song. You should, for example, have a different method for each verse of the song.

You are not allowed to use more advanced features than what we have covered in class. For this assignment, you should limit yourself to the Java features covered in Chapter 1 of the text.

You should include a comment at the beginning of your program with some basic information and a description of the program, such as:

```
// Suzy Student
//
// CSE 142, Summer 2005
// Section AE (TA: Kate)
// Programming Assignment #1
// 6/28/2005
//
// This program will ...
```

Submission and Grading:

Name your file `Song.java` and turn it in electronically from the "Assignments" link on the course web page.

Part of your program's score will come from its "external correctness." External correctness measures whether the output matches exactly what is expected. (We are picky about the output matching exactly!)

The rest of your program's score will come from its "internal correctness." Internal correctness measures whether your source code follows the stylistic guidelines specified in this document. This includes having an adequate comment header and capturing the structure and redundancy of the song as specified previously.