# CSE 142, Spring 2008
# Programming Assignment #1: Song (10 points)
### Due Tuesday, April 8, 2008, 4:00 PM

## Program Description:

This program tests your understanding of static methods and `println` statements. Write a Java class called `Song` in a file named `Song.java`. (Use exactly this file name, including identical capitalization.) Your program should produce the following song as output:

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

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

There was an old woman who swallowed a horse,
She's dead, of course!
```

The output is six verses of one version of the nursery rhyme “There Was an Old Woman Who Swallowed a Fly”. For brevity, we reduced the number of verses to six. Your program must **exactly** reproduce the output above. This includes identical wording, spelling, spacing, punctuation, and capitalization. You may include blank lines at the very end of the output if you like.

## Stylistic Guidelines:

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.

You should not place any `println` statements in your `main` method. (It is okay for `main` to have empty `println` statements to print blank lines.) Instead of printing in `main`, use static methods for two reasons:

1. <u>To capture the *structure* of the song's verses.</u>

You should write static methods to capture the structure of the song. You should, for example, have a method for each verse of the song to print that verse's entire contents. You can write additional methods as you see fit.

2. <u>To avoid simple *redundancy* in the output.</u>

You should use only one `println` statement for each distinct non-blank line of the song. For example, the following line appears several times in the output, but you should have only one `println` statement in your program that prints that line of the song:

```
I don't know why she swallowed the fly,
```

However, a method that prints just one line is not very useful. Instead, identify groups of lines that appear in multiple places in the song and create methods that capture those groups and are called multiple times. There is a general cumulative structural redundancy to the song that you should eliminate with your methods. Recall that methods can call other methods if necessary (which can themselves call other methods, and so on). The key question to ask is whether you have repeated lines of code that could be eliminated if you structured your methods differently. This includes sequences of `println` statements and repeated sequences of method calls.

Include a comment at the beginning of your program with some basic information and a description of the program in your own words. For example:

```
// Suzy Student, CSE 142, Autumn 2049, Section XX
// Programming Assignment #1, 06/07/49
//
// This program's behavior is ...
```

For this assignment, you should limit yourself to the Java features covered in Chapter 1 of the textbook. Though we will cover Chapter 2 while you work on this assignment, please do not use Chapter 2 features such as `print` statements (as opposed to `println`).

As a point of reference, our solution to this program has 10 methods other than `main` and occupies 73 lines including comments and blank lines. But this is just a rough guideline; you <u>do not</u> have to match this exactly.

## Submission and Grading:

Turn in your `Song.java` file electronically from the Homework 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 <u>exactly</u> what is expected. (We are *very picky* about the output matching exactly. Every character and space must match.) Programs that do not compile will receive no external correctness points.

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. You should also limit the lengths of all lines in your program to fewer than 100 characters.