Computer Science, Claremont McKenna College
CS51.1 - Introduction to Computer Science, Spring 2015

Problem Set 1 Due: 11:55 AM, Jan 29, 2015

General Instructions

Please carefully read and follow the directions exactly for each problem. Files and classes should be named exactly as directed in the problem (including capitalization!) as this will help with grading.

You should create your programs using your preferred text-editor, the Eclipse text editor, or jGrasp. Do not use a word processor such as Microsoft Word, WordPad, Google Docs, Apple's Pages, etc...

Your programs should be formatted in a way that's readable. In other words, indent appropriately, use informative names for variables, etc… If you are uncertain about what is a readable style, look at the examples from class as a starting point for a reasonable coding style.

This problem set assumes you have installed Java and an IDE (Eclipse or jGrasp) on your system. Please see the course web page on Sakai for installation instructions.

Your programs should compile and run without errors. Please do not submit programs that do not compile! It's better to submit partial implementation that compiles as oppose to implementations that do not compile.

At the top of each file you submit, you should put your name and your email as comments.

This homework set should be submitted via Sakai using the Assignment Menu option (on the left pane). You should only submit the requested ".java" files. Do not submit ".class" files or Eclipse-specific project files. Finally, please do not submit files using Sakai's Dropbox Menu.

Turn in the following file(s):

```
SixDays.java
```

Problem 1

In this first assignment, you will practice writing simple static methods and println statements. You should write a Java class called `SixDays` that should be saved into a file called `SixDays.java`. Your program should produce the following song as output:

```
On the 1st day of "Xmas", my true love sent to me
a partridge in a pear tree.

On the 2nd day of "Xmas", my true love sent to me
two turtle doves, and
a partridge in a pear tree.

On the 3rd day of "Xmas", my true love sent to me
three French hens,
two turtle doves, and
a partridge in a pear tree.

On the 4th day of "Xmas", my true love sent to me
four calling birds,
three French hens,
two turtle doves, and
a partridge in a pear tree.

On the 5th day of "Xmas", my true love sent to me
five golden rings,
four calling birds,
three French hens,
two turtle doves, and
a partridge in a pear tree.

On the 6th day of "Xmas", my true love sent to me
six geese a-laying,
five golden rings,
four calling birds,
three French hens,
two turtle doves, and
a partridge in a pear tree.
```

The song is a modified version of a classic holiday song. For brevity, we reduced the number of days from the original twelve to six and shortened Christmas to "Xmas".

You should **exactly** reproduce the format of this output. This includes having identical wording, spelling, spacing, punctuation, and capitalization. Please do not include additional verses, such as writing twelve days to match the complete song. You may include a blank line at the very end of the output if you like.

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.

Guidelines:

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. To capture the *structure* of the song's six verses.

You should write static methods to capture the structure of the song. You should, for example, have a method for each of the six verses of the song to print that verse's entire contents.

2. To avoid simple *redundancy* in the output.

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:

```
a partridge in a pear tree.
```

However, a method that prints a single line such as the above is not useful. Instead, you should identify groups of two or more lines that appear in multiple places in the song and create static methods that capture those groups and are called multiple times. There is a general structural redundancy to the song that you should eliminate with your static methods. Recall that methods can call other methods if necessary. 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. As a point of reference, our solution to this program has twelve static methods other than `main` and occupies 89 lines including comments and blank lines.

You do NOT have to eliminate redundancy in lines that are similar but not identical, such as these:

```
On the 1st day of "Xmas", my true love sent to me
On the 2nd day of "Xmas", my true love sent to me
```

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.

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 mathematical expressions, `print` statements (as opposed to `println`), or `for` loops on this program.