CMPSC 111
Introduction to Computer Science I
Fall 2016

Practical 7
November 4, 2016
Due in Bitbucket by midnight of the day of your practical
"Checkmark" grade

Summary

To review for the upcoming examination, in this practical you will create and then call methods that determine if certain events occurred during the year given by the user. For this task you will use `if/else` statements and boolean logic operators such as "`&&`" or "`||`". Finally, using the "`git add`", "`git commit`", and "`git push`" commands you will upload your modified source code files and the output you obtain from running your program to your Git repository hosted by Bitbucket.

Review the Textbook

You may refer to sections 5.1–5.3 in your textbook to further review `if/else` statements and boolean expressions. Please examine Chapter 4 if you have questions about modifying Java classes.

Save the Example Programs

Get the files "`YearChecker.java`" and "`YearCheckerMain.java`" from the shared course Git repository. Using the technique from a previous assignment, copy these files into your own Bitbucket repository into a directory called "`practical08/`". Study these programs first and make sure you understand them. Please note that these programs are not complete and will not run correctly.

A Program Inspired by "Through All the Years"

This section's title is a quotation is from Allegheny College's Alma Mater. In one of the classes for this assignment, `YearChecker.java`, you will write methods that, for the user's input from `YearCheckerMain.java`, determine which of the following events occurs that year:

- it is designated as a leap year
- the emergence of the 17-year cicadas (more specifically, Brood II)
- it is predicted to be a peak year of sunspot activity

A year is a *leap year* if it is divisible by 4, *unless* it is a century year. If it is a century year, it is a leap year if it is divisible by 400. For instance, 1968 and 1972 are leap years since they are divisible by 4; 1967 and 1970 are not. The year 2000 is a leap year because it is divisible by 400; however, 1900 is not (even though 1900 is divisible by 4—century years are treated differently).

The 17-year cicadas emerge from underground every 17 years. There are several "broods"—the one in which we have interest emerged most recently in 2013. That is, any year that differs from 2013 by a multiple of 17 is also an emergence year for Brood II (e.g., 2040, 1996, 1928, and 3713).

Sunspot activity usually peaks every 11 years. The year 2013 was supposed to be such a "solar max" year. This means that any year that differs from 2013 by a multiple of 11 should also be a solar max year. For instance, 2002, 2024, and 1793 are all years predicted to exhibit peak activity.

Here is the sample output from running the program with the year 1452 as input:

```
aldenv112:practical7 gkapfham$ java YearCheckerMain

Enter a year between 1000 and 3000: 1452
1452 is a leap year
It's an emergence year for Brood II of the 17-year cicadas
It's a peak sunspot year.
Thank you for using this program!
```

In addition to regularly compiling your program, you should thoroughly test it as well! Try the above date, the other dates mentioned in the assignment, and some of your own as well.

Modify the Programs

- Edit the file `YearChecker.java` and find the constructor in this class. Make sure you understand what the constructor initializes, when it gets called, what method calls it, and what happens after it is executed. Please ask the instructor or the teaching assistants if you are not clear about the purpose and behavior of the constructor provided by this class.
- Edit the file `YearChecker.java` and find the three methods with the names `isLeapYear()`, `isCicadaYear()`, and `isSunspotYear()`. You task for this practical is to implement the conditional logic that these methods need to execute, as described in the previous section.
- Improve the file `YearCheckerMain.java` by adding method calls to `isLeapYear()`, `isCicadaYear()`, and `isSunspotYear()`, or in other words, by writing Java statements that will invoke these three methods. You should aim to produce output like that given above.

Completing the Practical Assignment

To finish this assignment and earn a "checkmark", you should submit the `YearChecker.java` and `YearCheckerMain.java` files you edited in your Bitbucket repository by using appropriate `git` commands. You also need to submit an output file with at least five runs of your program.

General Guidelines for Practical Sessions

- **Submit *Something*.** Your grade for this assignment is a "checkmark" indicating whether you did or did not complete the work and submit something to the Bitbucket repository using the "`git add`", "`git commit`", and "`git push`" commands.
- **Update Your Repository Often!** You should `add`, `commit`, and `push` your updated files each time you work on them, always including descriptive messages about each code change.
- **Review the Honor Code Policy on the Syllabus.** Remember that while you may discuss your work with other students in the course, code that is nearly identical to, or merely variations on, the work of others will be taken as evidence of violating the Honor Code.