CSC 20: Project 5 - REPL

This semester we are going to build a simple logic simulator. The steps to using the simulator will go something like this:

~ Users of the simulator will specify a boolean logic formula using text.
~ The simulator will build an equivalent logic circuit in memory.
~ The user of the simulator will specify input values for the circuit.
~ The simulator will evaluate the circuit and print the result.

In this step of the project, you will write a simple program that allows users to specify a logical expression and evaluate it for various inputs.

A REPL (read-evaluate-print-loop) is a program that issues a prompt, waits for the user to type some input, evaluates the input, and prints its response. It does this in a loop until the user is done. Project 5 should be a REPL that behaves like this:

```
Welcome to Circuit Master 2000
> A=T
error
> A*T
false
> A=T
true
> B=F
error
> (A+-A)
true
> A*B*C
false
> A=T
false
> B=T
false
> C=T
true
> exit
Goodbye!
```

In this example, the "> " is the prompt (your program should print this). What comes after the prompt is whatever is typed by the user, and the `true`, `false`, and `error` are produced by your program.

Specification:

In a file named `REPL.java` write a program that implements your REPL. It should be fairly short because most of the hard work is done in other files.

Your REPL should print a welcome when it is started. It should then go into a simple loop.

```
Print welcome message
While not done
   Print a prompt
   Read a line
   Process a line
Print goodbye message
```

Most of the action is in the processing of lines. There are three types of allowable input.

Exit command. An input line that contains "exit" should terminate the loop.

Variable assignment. A line that contains an equals character `'='` is a variable assignment. Left of the equals should be a single upper-case letter variable (not 'T' or 'F'). Right of the equals should be 'T' or 'F'. The result of this command should be to update the value associated with the variable in the current circuit's Map to `true` or `false`. The current circuit should then be reevaluated and its result should be output as `true` or `false`. If the variable does not exist in the current circuit's Map or no circuit has yet been defined, output `error`.

A new logic expression. If the input line has no equals and is not "exit", then it should be assumed to be a new logic expression. Any existing circuit and its variable map should be disregarded and a new one should be built based on the new expression. The circuit should then be evaluated and its result should be output as `true` or `false`. Note that our gates are all defined to have their inputs default to `false`, so the initial circuit value will be with all inputs `false`.

Notes:

~ There will be no spaces in any inputs.

~ All variables will be single upper-case letters, except 'T' and 'F' cannot be variables.

~ All expressions will follow the required format of Project 4.

~ You do not need to handle errors. All inputs will be valid (except for the errors listed in the specification).

~ Use Scanner's `nextLine` to get each user input.

~ Your program should print nothing other than what is specified.

~ You will not get multiple attempts to get this project working correctly, so test thoroughly before submission.

To Receive Credit:

Submit only your file `REPL.java`. I will compile your code in a directory with my version of all supporting files (so make sure your `REPL.java` is only using the public methods specified in prior projects).

Follow the directions in Project Requirements and DBInbox Submission, and submit by 11:59pm, Friday, May 19, 2017.

Questions?

If something is not clear, ask questions in class, on Piazza, or in office hours. Do not wait until the last minute to clear things up. Start early!

first published 12pm, May 2, 2017