Write a Java program, in file Logic.java that can be run as follows, assuming that there is a file called f in your current directory:

[ID@serv15 project]$ java Logic f

When your program is run, it should read a description of a network of logic gates and wires from the file f (or from any other file you elect to use). Here is an example description:

gate A and 1.0

gate B or 0.5

wire A out B in1 0.1

wire B out A in1 0.3

-- another gate

gate C neg 0.5

wire A out C in 0.1

wire C out B in2 0.1

For gates, the arguments in the source file are:

* The gate name (textual, even numbers are legal)
* The gate type (textual, for now, this has no meaning)
* The gate's time delay (a positive floating point number)

For wires, the arguments are:

* The wire source (a gate name)
* The source pin name (textual, for now, this has no meaning)
* The wire destination (a gate name)
* The destination pin name (textual, for now, this has no meaning)
* The wire delay (a positive floating point number)

Note in the above that we don't use punctuation to indicate a pin of a gate, for example, A.out, we just use a space.

A significant part of this assignment is that your program must tolerate and appropriately report errors in the input. That is, numbers must be given where expected, no gate name may be defined more than once, all gate names referenced in wire descriptions must have been declared before use, no input pin of a gate may also be used as an output, and no input pin may be used more than once.

If and only if there are no errors in the input, your program should generate as output to standard output (System.out) a reconstruction of the the input, although the order of gates and wires in the output may be changed, spacing may be standardized, and comments may be lost. Furthermore, numeric fields may be output in equivalent form — so for example, 05.6 or 5.600 might bothe end up being output as 5.6.

Error messages should be output to standard error (System.error) and should reasonably document the error, for example:

cannot open file named xyzzl.bug

shoe found where gate or wire expected

gate X illegally redefined

wire X out Y in undefined source

wire X Z delay expected

When there are errors, the output to standard output should include all correct parts of the input and, if it includes incorrect parts, those should make sense. The format of the error messages and the output of incorrect parts of the input are deliberately underspecified. You have design decisions to make here.

Note: For now, the count of the number of inputs permitted on a gate is just a number. Later, we will add code to enforce this: A 3 input gate must have exactly 3 wires to its inputs, and we will eliminate this field on gates that only have one input (neg, for example). If you want extra work, try enforcing this.

**Grading criteria:** No credit will be given to programs that do not represent a substantial attempt at meeting the requirements of this assignment. Your program must:

* Begin with a header comment giving the source file name.
* Contain Javadoc comments claiming authorship and declaring the version to be MP1.
* Be cleanly and readably indented.
* Use appropriate variable names.
* Respond appropriately to errors in the input.
* Not demonstrate any unhandled or mishandled exceptions.
* Properly reconstruct the correct part of the input.

Note that MP3 will involve extending MP2 to support gates with specific named inputs and outputs, so planning for this, while not part of this assignment, may simplify the next.

Note that the requirements for header comments in the file header are absolute. Your name must appear in the form that it appears on your ID card. The TAs will not waste their time doing detective work to attempt to figure out who submitted what code.

**Submission:** Your solution should consist of a single file named Logic.java, with one public method, main. (We will chop the file into smaller more managable pieces later, as it grows.)