Program: Infix to Postfix (+ Evaluator)

Write a program that converts infix expressions to their postfix equivalents and subsequently evaluates them. Receive input from `stdin` and send output to `stdout`. An example run of your program might be: `java InfixToPostfix < input.txt > output.txt`. The input contains a single infix expression per line as follows:

```
5+7
3*8+6
2^2+9
7^7-6^6
9-5*8/4^2
2^3^2-4^2+4
3*3+9*8-2
6-4-3+3+4*6/2+8*8
4-3*(4*(3*6-(8+9)^2)*(9-3)/3)-7
(3+(9-3)*(5-3)*4)
((((((9-9))))))
((9/2*3/2+1)*1)*1*1*1*1
9*3-(5-7/5)
```

There are several requirements:

- You must use your generic `Stack` and `Queue` classes to implement the infix to postfix conversion and the subsequent evaluation. This will, in turn, mean using your generic `List` and `Node` classes. To make this all work, make sure that `Stack.class`, `Queue.class`, `List.class`, and `Node.class` are in the same directory as your infix to postfix source (e.g., `InfixToPostfix.java`).
- Structure your output so that it displays the infix expression on a line, the postfix expression on the next line, the result on the next line, followed by a blank line. For example:

```
5 + 7
5 7 +
12.0

9 * 3 - ( 5 - 7 / 5 )
9 3 * 5 7 5 / - -
23.4
```

- Evaluations must be precise (i.e., no integer division). Think about what this means in terms of your evaluation stack.
- Note that `2^3^2 = 512`!