```java
import components.queue.Queue;

/**
 * {@code Queue} represented as a {@code Sequence} of entries, with
 * implementations of primary methods.
 *
 * @param <T>
 *            type of {@code Queue} entries
 * @correspondence this = $this.entries
 */
public class HelloWorld {

    public static void main(String[] args) {

    }

    /**
     * Evaluates a Boolean expression and returns its value.
     *
     * @param tokens
     *            the {@code Queue<String>} that starts with a bool-expr string
     * @return value of the expression
     * @updates tokens
     * @requires [a bool-expr string is a prefix of tokens]
     * @ensures <pre>
     * valueOfBoolExpr =
     *   [value of longest bool-expr string at start of #tokens] and
     * #tokens = [longest bool-expr string at start of #tokens] * tokens
     * </pre>
     */
    public static boolean valueOfBoolExpr(Queue<String> tokens) {
        boolean answer = false;
        while (tokens.length() > 0
                && !tokens.front().equals("### END OF INPUT ###")) {
            String front = tokens.dequeue();
            switch (front) {
                case "T": {
                    answer = true;
                    break;
                }
                case "F": {
                    answer = false;
                    break;
                }
                case "NOT": {
                    answer = !valueOfBoolExpr(tokens);
                    break;
                }
                case "(": {
                    answer = valueOfBoolExpr(tokens);
                    break;
                }
                case ")": {
                    break;
                }
                case "AND": {
                    answer &= valueOfBoolExpr(tokens);
                    break;
                }
                case "OR": {
                    answer |= valueOfBoolExpr(tokens);
                    break;
                }
                default:
                    break;
            }
        }
        return answer;
    }

}
```