Here is a summary of the conventional syntax (latex) in terms of the official syntax of the Nqthm logic.

1. Variables are printed in italics, unless specified otherwise in the table below.

2. Function application. For any function symbol for which special syntax is not given below, an application of the symbol is printed with the usual notation; e.g., the term `(fn x y z)`is printed as $\mathrm{fn}\,(x,\ y,\ z)$. Note that the function symbol is printed in Roman. In the special case that 'c' is a function symbol of no arguments, i.e., it is a constant, the term `(c)`is printed merely as c, in small caps, with no trailing parentheses. Because variables are printed in italics, there is no confusion between the printing of variables and constants.

3. Other constants. Quoted constants are printed in the ordinary syntax of the Nqthm logic, in a 'typewriter font.' For example, `'(a b c)` is still printed just that way. `#b001` is printed as $\mathtt{001}_2$, `#o765` is printed as $\mathtt{765}_8$, and `#xa9` is printed as $\mathtt{a9}_{16}$, representing binary, octal and hexadecimal, respectively.

4. `(if x y z)` is printed as

   if x **then** y **else** z **fi**.

5. `(cond (test1 value1) (test2 value2) (t value3))` is printed as

   if *test1* **then** *value1* **elseif** *test2* **then** *value2* **else** *value3* **fi**.

6. `(case x (key1 answer1) (key2 answer2) (otherwise default))` is printed as

   case on x: **case** = `key1` **then** *answer1* **case** = `key2` **then** *answer2* **otherwise** *default* **endcase**.

7. `(let ((var1 val1) (var2 val2)) form)` is printed as

   let *var1* **be** *val1*, *var2* **be** *val2* **in** *form*.

8. `(let* ((var1 val1) (var2 val2)) form)` is printed as

   let* *var1* **be** *val1*, *var2* **be** *val2* **in** *form*.

9. `(forall (x y) (p x))` is printed as

   $\forall\, x,\ y\colon \mathrm{p}\,(x)$.

10. `(exists (x y) (p x))` is printed as

    $\exists\, x,\ y\colon \mathrm{p}\,(x)$.

11. `(not x)` is printed as

    $\neg\ x.$

12. The remaining symbols that are printed specially are described in the following table.

Nqthm Syntax	Conventional Syntax
`t`	$\mathbf{t}$
`f`	$\mathbf{f}$
`nil`	$\mathbf{nil}$
`(union-theories x y)`	$x \cup y$
`(set-difference-theories x y)`	$x\ \mathbf{less}\ y$
`(intersection-theories x y)`	$x \cap y$
`(congruent x y)`	$x \cong y$
`(or x y)`	$x \vee y$
`(and x y)`	$x \wedge y$
`(* x y)`	$x * y$
`(- x y)`	$x - y$
`(+ x y)`	$x + y$
`(union x y)`	$x \cup y$
`(remainder x y)`	$x\ \mathbf{mod}\ y$
`(/ x y)`	$x\ /\ y$
`(iff x y)`	$x \leftrightarrow y$
`(implies x y)`	$x \rightarrow y$
`(append x y)`	$x\ @\ y$
`(member x y)`	$x \in y$
`(>= x y)`	$x \geq y$
`(geq x y)`	$x \geq y$
`(> x y)`	$x > y$
`(greaterp x y)`	$x > y$
`(<= x y)`	$x \leq y$
`(leq x y)`	$x \leq y$
`(e0-ord-< x y)`	$x \leq_\epsilon y$
`(< x y)`	$x < y$
`(lessp x y)`	$x < y$
`(equal x y)`	$x = y$
`(= x y)`	$x =_n y$
`(eq x y)`	$x =_{eq} y$
`(not (member x y))`	$x \notin y$
`(not (>= x y))`	$x \not\geq y$
`(not (geq x y))`	$x \not\geq y$
`(not (> x y))`	$x \not> y$
`(not (greaterp x y))`	$x \not> y$
`(not (<= x y))`	$x \not\leq y$
`(not (leq x y))`	$x \not\leq y$
`(not (e0-ord-< x y))`	$x \not\leq_\epsilon y$
`(not (< x y))`	$x \not< y$

Nqthm Syntax	Conventional Syntax
`(not (lessp x y))`	$x \not< y$
`(not (equal x y))`	$x \neq y$
`(not (= x y))`	$x \neq_n y$
`(not (eq x y))`	$x \neq_{eq} y$
`(minus x)`	$-\ x$
`(1+ x)`	$1 + x$
`(zerop x)`	$x \simeq \mathtt{0}$
`(numberp x)`	$x \in \mathbf{N}$
`(sub1 x)`	$x - 1$
`(not (zerop x))`	$x \not\simeq \mathtt{0}$
`(not (numberp x))`	$x \notin \mathbf{N}$