```
// ================================================================
// RISCALDemo.txt
// A demonstration of RISCAL (c) 2018, Wolfgang Schreiner
// https://www.risc.jku.at/research/formal/software/RISCAL/
// See file "README.txt" for how to run the course exercises.
// ================================================================

// symbol input (optional): type string and then Ctrl+# ~> symbol

// a constant and the type of the natural numbers 0..N
val N = 8;
type Num = ℕ[N]; // Nat ~> ℕ, Int ~> ℤ

// a function
fun muladd(a:Num, b:Num, c:Num):ℕ[N·N+N] =
  a·b+c // a term (* ~> ·)
;

// a predicate
pred input(m:Num, n:Num) ⇔
  n ≠ 0 // a formula (~= ~> ≠, <= ~> ≤, >= ~> ≥)
;

// another predicate
pred output(m:Num, n:Num, q:Num) ⇔
  // ~ ~> ¬, /\ ~> ∧, \/ ~> ∨, => ~> ⇒, <=> ~> ⇔
  // forall ~> ∀, exists ~> ∃
  ∃r:Num. m = muladd(n,q,r) ∧
    ∀r0:Num, q0:Num. m = muladd(n,q0,r0) ⇒ r ≤ r0
;

// a predicate that is expected to be true
theorem satisfiable() ⇔
  ∀m:Num,n:Num. input(m,n) ⇒ ∃q:Num. output(m,n,q)
;

// an alternative formulation of the same predicate
theorem satisfiable0(m:Num,n:Num)
  requires input(m,n);
⇔ ∃q:Num. output(m,n,q)
;

// an implicitly defined function
fun compute(m:Num, n:Num): Num
  requires input(m,n);
= choose q:Num with output(m, n, q);

// ================================================================
// end of file
// ================================================================
```