

# Logic as a Path to Enlightenment

## Work in Progress Report

Wolfgang Schreiner

Research Institute for Symbolic Computation (RISC)  
Johannes Kepler University, Linz, Austria  
Wolfgang.Schreiner@risc.jku.at

# Enlightenment and Education

- ▶ **Enlightenment:** reject claims based on authority (“ipse [Aristotle] dixit”)
  - ▶ Only two sources of truth acceptable:
    - ▶ Empirical evidence (observation)
    - ▶ Well-formed arguments (reasoning).
  - ▶ Stark contrast to pre- or even anti-modern views.
- ▶ **Education:** often claims accepted by authority (“ipse [the teacher] dixit”)
  - ▶ Even in “rational” disciplines like mathematics or computer science.
    - ▶ Presentations of propositions, rules, methods, and algorithms (more often than not) lack proper justification.
  - ▶ Students educated to become “believers” (or, equally worse, “non-believers”) rather than “rational skeptics”.

Students should be provided a basis for rational discourse.

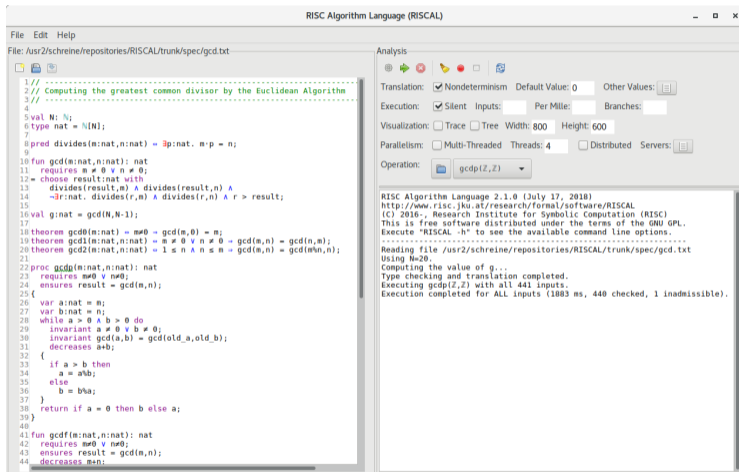
# Logic as a Path to Enlightenment

Logic as the “science of reasoning” provides such a basis.

- ▶ **Predicate logic:** the “modern” logic of today.
  - ▶ Starting with Frege’s “Begriffsschrift” in 1879.
  - ▶ Incorporates and supersedes Aristotle’s term logic.
  - ▶ Rich enough to capture most of mathematics and much of natural language.
- ▶ Construct **formal models of reality** with precise meaning and reasoning rules.
  - ▶ State propositions as formal sentences.
  - ▶ Derive valid arguments that prove the propositions.
  - ▶ Judge whether such arguments are valid or not.

Should be taught as a practical “working language” for modeling and reasoning.

# The RISCAL Software



The screenshot displays the RISCAL software interface. The main window is titled "RISCAL Algorithm Language (RISCAL)". The left pane shows a code editor with the following code:

```
1 // -----
2 // Computing the greatest common divisor by the Euclidean Algorithm
3 // -----
4
5 val N: N;
6 type nat = N[N];
7
8 pred divides(m:nat,n:nat) = ∃p:nat. m = p * n;
9
10 fun gcd(m:nat,n:nat): nat
11   requires m ≠ 0 ∧ n ≠ 0;
12 = choose result:nat with
13   divides(result,m) ∧ divides(result,n) ∧
14   ~∃r:nat. divides(r,m) ∧ divides(r,n) ∧ r > result;
15
16 val g:nat = gcd(N,N-1);
17
18 theorem gcd0(m:nat) = m = 0 → gcd(m,0) = m;
19 theorem gcd1(m:nat,n:nat) = m ≠ 0 ∧ n ≠ 0 → gcd(m,n) = gcd(n,m);
20 theorem gcd2(m:nat,n:nat) = 1 ≤ n ∧ n ≤ m → gcd(m,n) = gcd(m%n,n);
21
22 proc gcdp(m:nat,n:nat): nat
23   requires m ≠ 0 ∧ n ≠ 0;
24   ensures result = gcd(m,n);
25 {
26   var a:nat = m;
27   var b:nat = n;
28   while a > 0 ∧ b > 0 do
29     invariant a ≠ 0 ∧ b ≠ 0;
30     invariant gcd(a,b) = gcd(oid_a,oid_b);
31     decreases a+b;
32   {
33     if a > b then
34       a = a%b;
35     else
36       b = b%a;
37   }
38   return if a = 0 then b else a;
39 }
40
41 fun gcdf(m:nat,n:nat): nat
42   requires m ≠ 0 ∧ n ≠ 0;
43   ensures result = gcd(m,n);
44   decreases m+n;
```

The right pane shows the "Analysis" panel with the following settings:

- Translation:  Nondeterminism Default Value: 0 Other Values: [ ]
- Execution:  Silent Inputs: [ ] Per Mill: [ ] Branches: [ ]
- Visualization:  Trace  Tree Width: 800 Height: 600
- Parallelism:  Multi-Threaded Threads: 4  Distributed Servers: [ ]
- Operation: [ ] gcdp(Z,Z) [ ]

The bottom of the analysis panel shows the execution log:

```
RISCAL Algorithm Language 2.1.0 (July 17, 2018)
http://www.risc.jku.at/research/formal/software/RISCAL
(C) 2016-, Research Institute for Symbolic Computation (RISC)
This is free software distributed under the terms of the GNU GPL.
Execute "RISCAL -h" to see the available command line options.
-----
Reading file /usr2/schreine/repositories/RISCAL/trunk/spec/gcd.txt
Using N=20.
Computing the value of g...
Type checking and translation completed.
Executing gcdp(Z,Z) with all 441 inputs.
Execution completed for ALL inputs (1883 ms, 440 checked, 1 inadmissible).
```

Automatic checking of theorems, algorithms, and verification conditions.

# Conclusions

- ▶ Goal: logic-based **self-directed learning**
  - ▶ Teacher become “enablers” by providing basic knowledge and skills
  - ▶ Students “educate themselves” by solving problems.
    - ▶ (Voluntary) quizzes, (mandatory) assignments, possibly (graded) exams.
- ▶ Initial target: undergraduate university students.
  - ▶ Reachout both “up and down” to graduate students and to high-school students.
- ▶ Initial focus: computer science and mathematics.
  - ▶ First own courses on “Logic”, “Formal Modeling”, “Formal Methods”; later also others’ introductory courses on algorithms and software development.

Towards “enlightenment” via “rational thinking” by “self-directed learning”.