

# Demo: AlgoVer

Institut für Theoretische Informatik

sumAndMax/Bounds.1

! Edit

sumAndMax/Null.1

✓ Edit

sumAndMax/Bounds.2

✓ Edit

sumAndMax/Bounds.3

✓ Edit

sumAndMax/Null.2

✓ Edit

sumAndMax/InitInv

✓ Edit

sumAndMax/InitInv.1

! Edit

sumAndMax/InitInv.2

! Edit

sumAndMax/loop/else/Inv

✓ Edit

sumAndMax/loop/else/Inv.1

✓ Edit

sumAndMax/loop/else/Inv.2

✓ Edit

sumAndMax/loop/else/Dec

✓ Edit

sumAndMax/loop/else/Bounds

✓ Edit

sumAndMax/loop/else/Bounds.1

✓ Edit

sumAndMax/loop/else/Null

✓ Edit

sumAndMax/loop/then/Inv

✓ Edit

sumAndMax/loop/then/Inv.1

✓ Edit

sumAndMax/loop/then/Inv.2

✓ Edit

sumAndMax/loop/then/Dec

✓ Edit

sumAndMax/loop/then/Bounds

✓ Edit

```

7  sum := a[0];
8  max := a[1];
9
10 var i: int := 1;
11 while (i < a.Length)
12   invariant 0 <= i && i <= a.Length
13   invariant forall k: int :: 0 <= k && k < i ==> a[k] <= max
14   invariant i * max >= sum
15   decreases a.Length - i
16 {
17   if (a[i] > max)
18   {
19     max := a[i];
20   }
21   sum := sum + a[i];
22   i := i + 1;
23 }
24 }
25

```

Interaction on:

- different levels of abstraction for interaction
- different representations of the same problem

Switch between levels and/or representations is necessary.

- program code
- specification
- proof representation/proof obligation
- proof guidance/interaction

- interaction on different representations
- hidden dependencies between representations
- context change cognitively challenging for the user
- missing interaction possibilities on representations

An interactive program verification system that allows implementing and researching different interaction concepts:

- integration of different representations as views
- integration of different interaction concepts
- seamless transition between views

The user is ...

- 1 ... able to use appropriate view at all times
- 2 ... can easily switch views without losing focus
- 3 ... is able to determine the results of costly actions before executing them