

Making formal verification accessible to all

What is formal verification?

```
2 + 2 = 4
```

```
if ¬true then A else B
= { applying ¬ }
  if false then A else B
= { semantics of the if statement }
  B
```

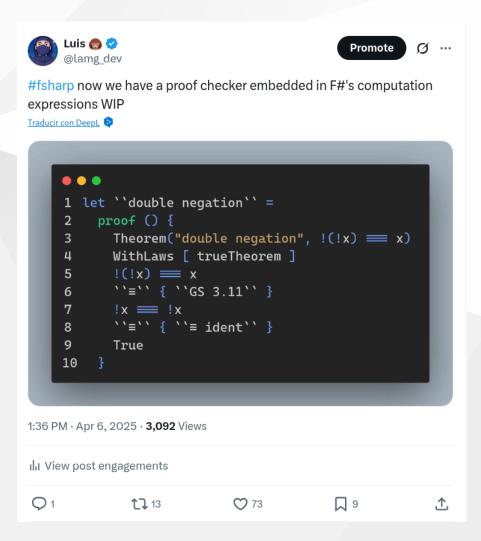
How can it be done

```
•••
1 +1-length : ∀ {ℓ}{A : Set ℓ}(h : A)(t : L A) → length (h :: t) ≡ length t + 1
2 +1-length h [] = refl
3 +1-length h t = refl
```

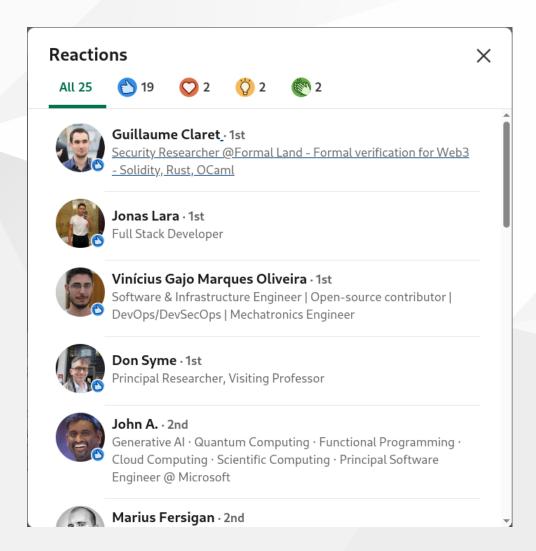
• Wybe, an accessible and powerful tool

```
1 proof {
2  theorem "double negation" (!(!x) == x)
3  !(!x) == x
4  ''='' { ''GS 3.11'' }
5  !x == !x
6  ''='' { ''= ident'' }
7  True
8 }
```

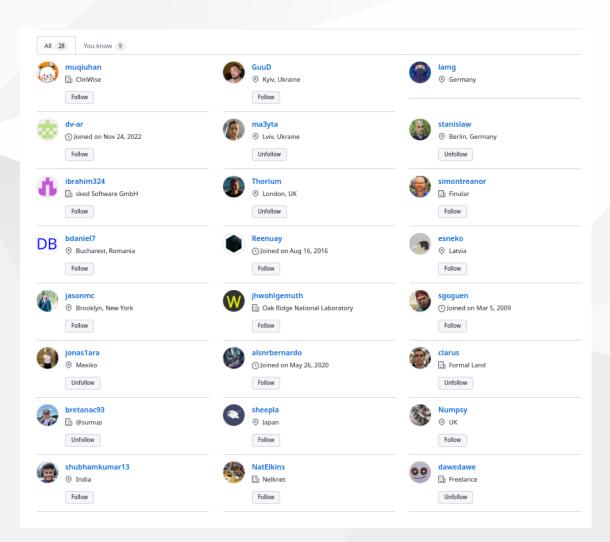
Post interactions X



Post interactions in LinkedIn



Stars in GitHub



The team

10 years ago

"Llegué a este estilo por mi preocupación de escribir los programas bien, pues los errores se me escapaban al escribir un programa y notaba que estaban ahí cuando los ejecutaba", afirma.



Luis Ángel Méndez, un bicho raro entre sus colegas. Foto: Eduardo González Martínez

El método Dijkstra se opone a la práctica imperante de escribir un programa primero y someterlo después a una corrección matemática. Con su propuesta, aunque más demorada, se obtiene un programa listo para ejecutarse, correcto desde su construcción.