

Wybe

**Making formal
verification
accessible to all**

What is formal verification?

- Mathematical formulas are universally true

$$2 + 2 = 4$$

Programs can be mathematical formulas

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

Zero program errors, but no 100% security

How can it be done




```
1 +1-length :  $\forall \{ \mathcal{Q} \} \{ A : \text{Set } \mathcal{Q} \} (h : A) (t : \mathbb{L} A) \rightarrow \text{length } (h :: t) \equiv \text{length } t + 1$   
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

**Luis** 
@lamg_dev

Promote  



#fsharp now we have a proof checker embedded in F#'s computation expressions WIP

[Traducir con DeepL](#) 

```
1 let ``double negation`` =  
2   proof () {  
3     Theorem("double negation", !(!x) == x)  
4     WithLaws [ trueTheorem ]  
5     !(!x) == x  
6     ``==`` { ``GS 3.11`` }  
7     !x == !x  
8     ``==`` { ``== ident`` }  
9     True  
10  }
```

1:36 PM · Apr 6, 2025 · 3,092 Views


 View post engagements


 1  13  73  9 


Post interactions in LinkedIn


Reactions


All 25

 19

 2


 2

 2




Guillaume Claret · 1st

Security Researcher @Formal Land - Formal verification for Web3 - Solidity, Rust, OCaml




Jonas Lara · 1st

Full Stack Developer




Vinícius Gajo Marques Oliveira · 1st

Software & Infrastructure Engineer | Open-source contributor | DevOps/DevSecOps | Mechatronics Engineer




Don Syme · 1st

Principal Researcher, Visiting Professor



John A. · 2nd

Generative AI · Quantum Computing · Functional Programming · Cloud Computing · Scientific Computing · Principal Software Engineer @ Microsoft



























Marius Fersigan · 2nd

8

Stars in GitHub

All 28

You know 9

 muqluhan ClinWise Follow	 GuuD Kyiv, Ukraine Follow	 lamg Germany Follow
 dv-ar Joined on Nov 24, 2022 Follow	 ma3yta Lviv, Ukraine Unfollow	 stanislav Berlin, Germany Unfollow
 ibrahim324 sked Software GmbH Follow	 Thorium London, UK Unfollow	 simontreanor Finlar Follow
 bdaniel7 Bucharest, Romania Follow	 Reenuay Joined on Aug 16, 2016 Follow	 esneko Latvia Follow
 jasonmc Brooklyn, New York Follow	 jhwohlgemuth Oak Ridge National Laboratory Follow	 sgoguen Joined on Mar 5, 2009 Follow
 Jonas1ara Mexico Unfollow	 alsnrbernardo Joined on May 26, 2020 Follow	 clarus Formal Land Unfollow
 bretanac93 @sumup Unfollow	 sheepla Japan Follow	 Numpy UK Follow
 shubhamkumar13 India Follow	 NatElkins Nelknet Follow	 dawedawe Freelance Unfollow

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.