

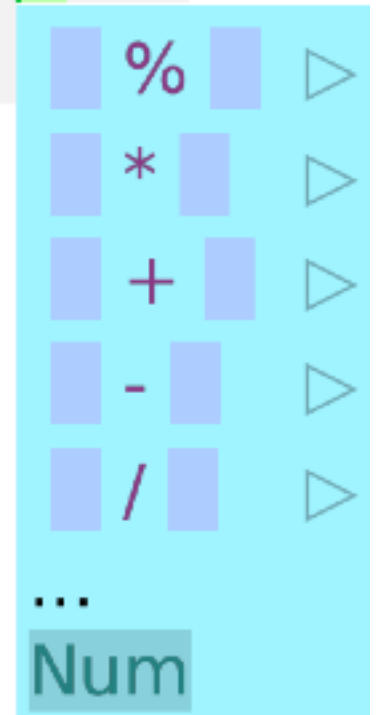


Structural Editing

```
>>> sum map 1 .. 10000  
      step 4  
      mapping x →
```

The AST is projected to a structured text presentation that is familiar to programmers.

The AST is always valid, but may be *incomplete*, with *holes* to fill in and *fragments* that contain subexpressions to transform.



Evaluation in Traditional REPLs

```
>>> sum(8./x/(x+2) for x in range(1,10000,4))  
3.141392653591789
```



Python's REPL (read-eval-print-loop) lets us play with our code interactively to verify our mental model of it.

Evaluation in Lamdu / Live Debugging

- Shows results for *all* subexpressions
- Code is evaluated while being typed
- Browse between different invocations of the same function

Annotations Evaluation Theme light

```
>>> sum map 1 .. 10000  
step 4  
[1, ...]  
mapping x → 8 / x / (x + 2)  
1 8 3  
2.6666666666666665  
[2.6666666666666665, ...]  
3.141392653591789
```