```haskell
sum [] = 0
sum (x:xs) = x + sum xs


isEven x = x `div` 2 == 0


sumEvens l =
  let evens = filter isEven
  in sum evens
```

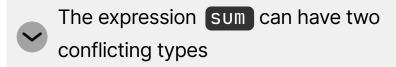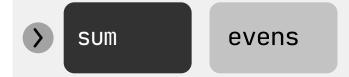The expression `sum` can have two conflicting types

> `sum`   `evens`

Conflicting types

Possible type 1

```haskell
sum :: [Int] → Int
```

Inferred from the orange highlights on the left side

Possible type 2

```haskell
sum :: ([Int]→[Int]) → Int
```

Inferred from the blue highlights on the left side

Relevant type information

```haskell
evens :: [Int] → [Int]
```

Inferred from the blue highlights on the left side