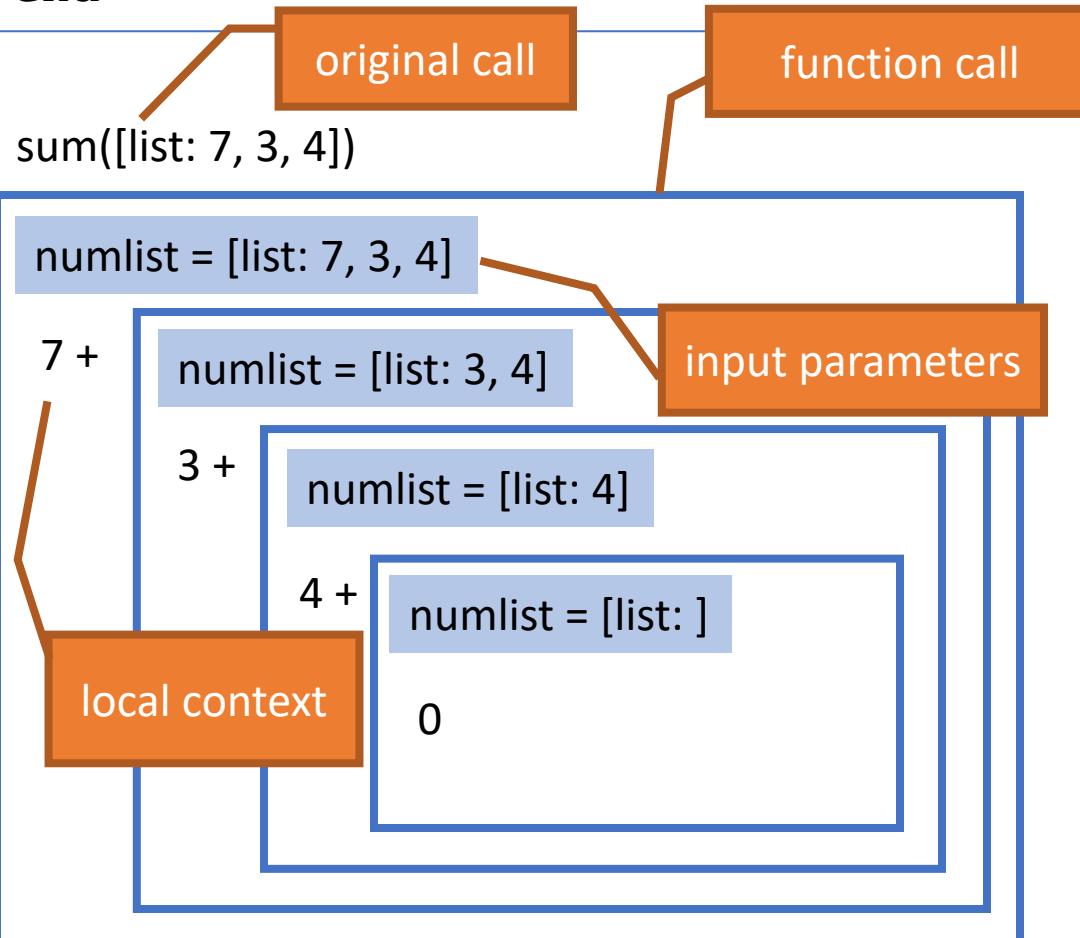


```

fun sum(numlist :: List<Number>) -> Number:
  doc: "sums numbers in list"
  cases (List) numlist:
    | empty => 0
    | link(fst, rst) => fst + sum(rst)
  end
end

```



This slide shows a **FUNCTION CALL DIAGRAM**.

It starts with a concrete example.

Each time a function gets called, we draw a box. The parameters are listed in the shaded area inside the box. When the function call finishes, we erase the box (and its contents).

The **local context** is the part of the computation that is waiting on the result of the function call.

```

fun sum(numlist :: List<Number>) -> Number:
  doc: "sums numbers in list"
  cases (List) numlist:
    | empty => 0
    | link(fst, rst) => fst + sum(rst)
  end
end

```

sum([list: 7, 3, 4])

numlist = [list: 7, 3, 4]

7 +

 numlist = [list: 3, 4]

3 +

 numlist = [list: 4]

4 +

 numlist = [list:]

local context

0

original call

function call

input parameters

Here's another view
with more detail

In the blocks, we draw a
nested box as []
(in the local context part)

original call

function call

input parameters

local context

