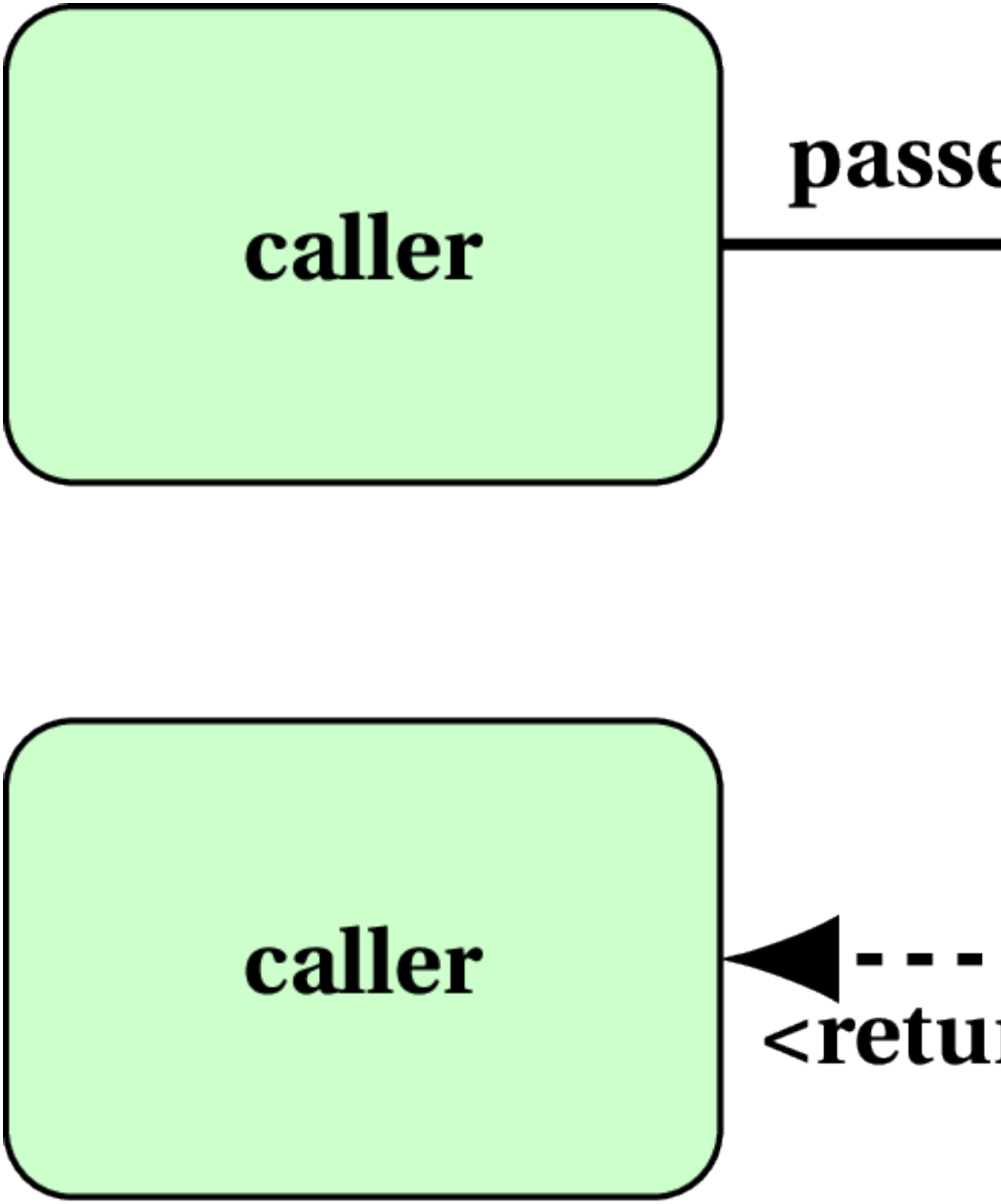
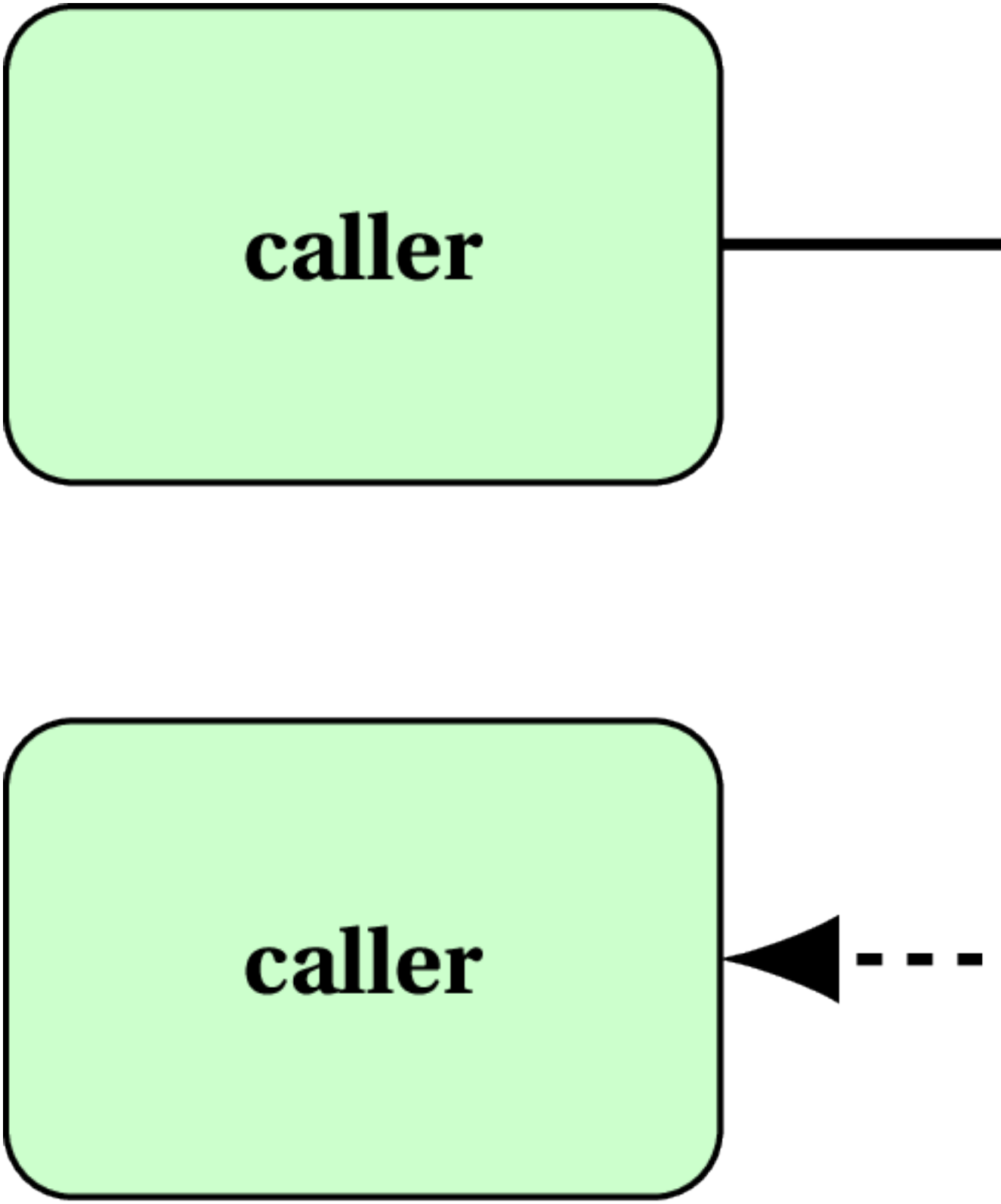
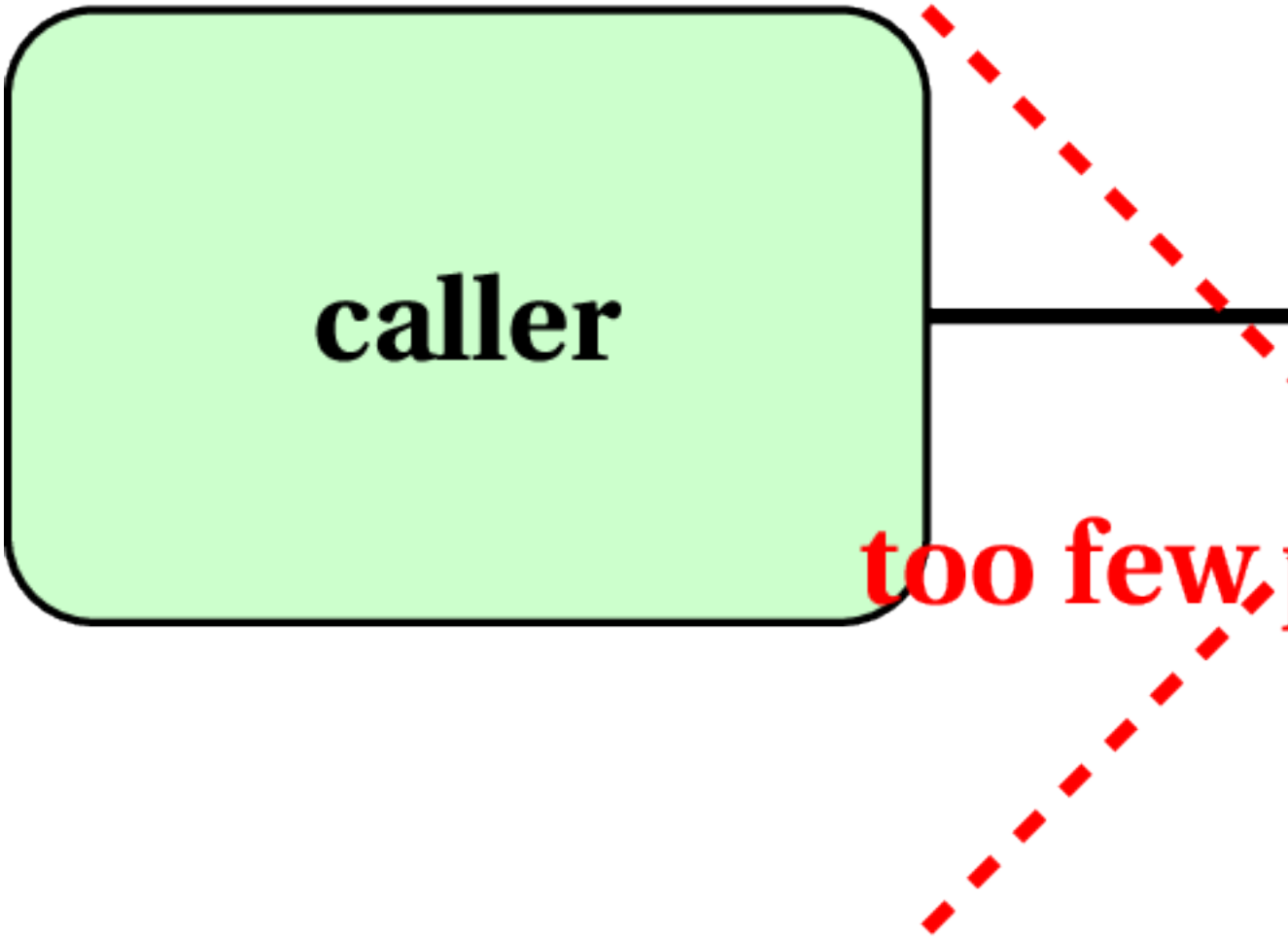

caller
passe
caller
- - -
<retu

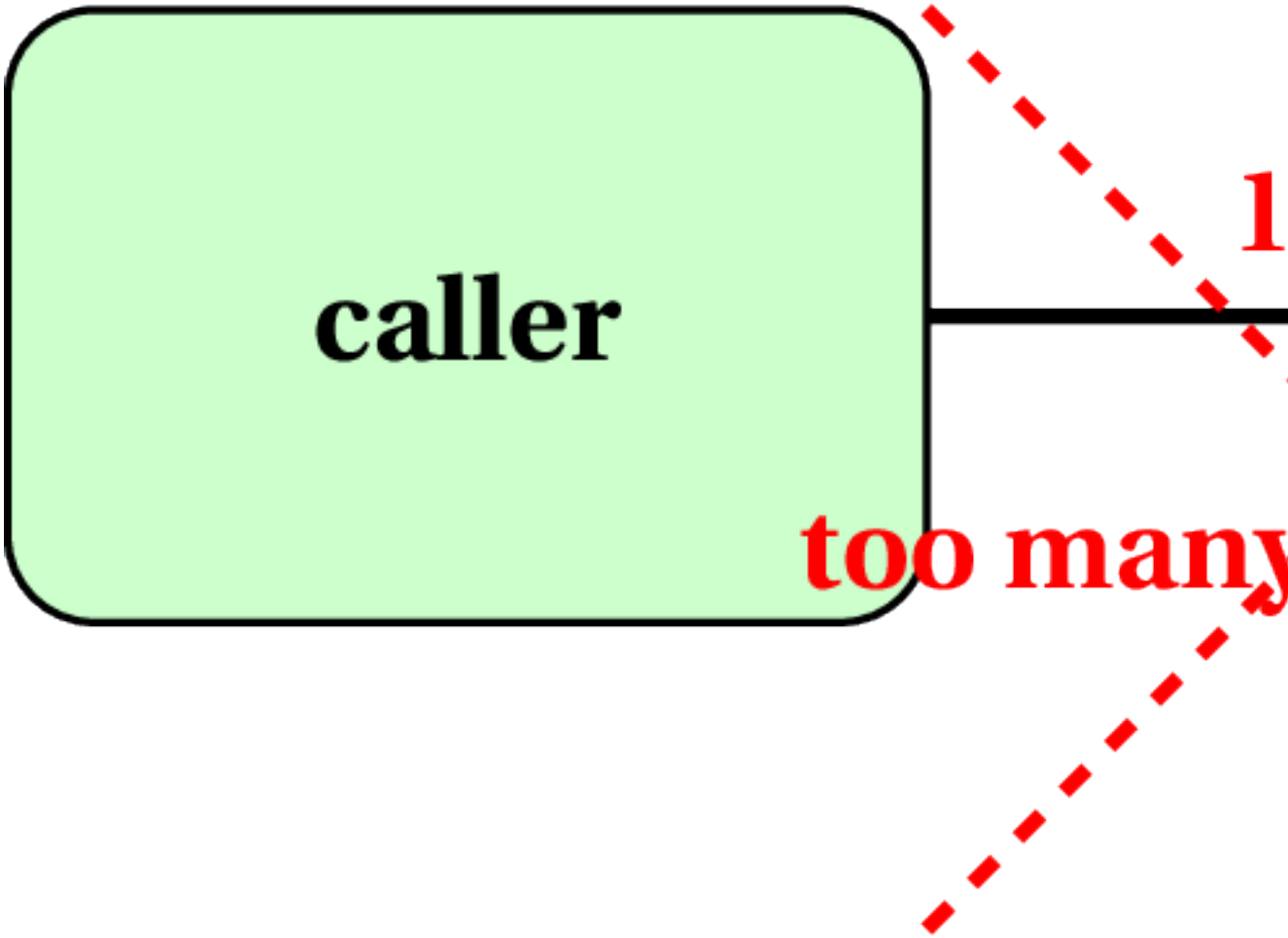
Example 1


caller
caller

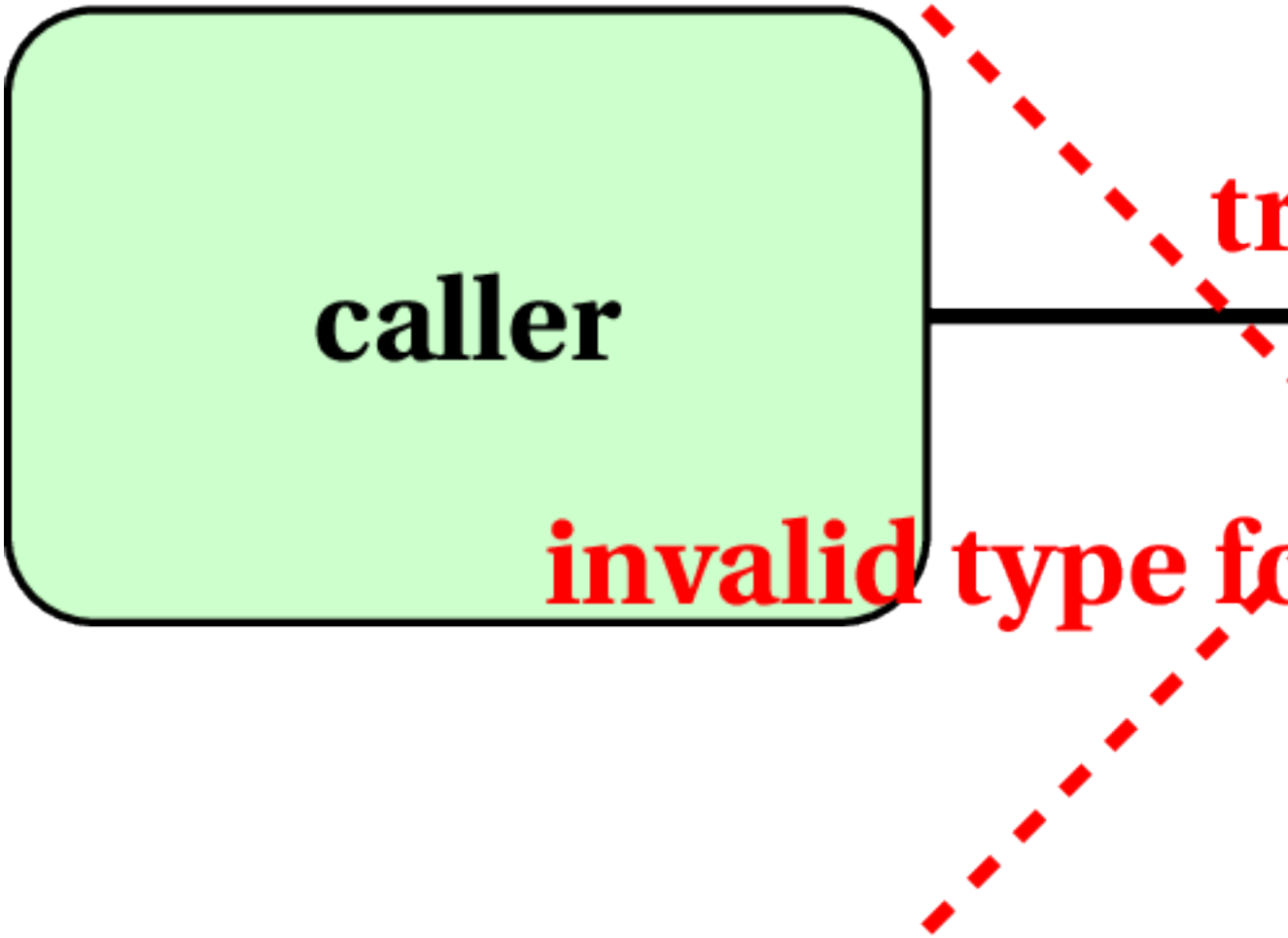
Example 2


caller
too few

Example 3


caller
too many

Example 4


caller
tr
invalid type fo

Syntax of a function is:

```
returnType function( <parameters>) {
    <some code>
    <return statement>
    <some code>
}
```

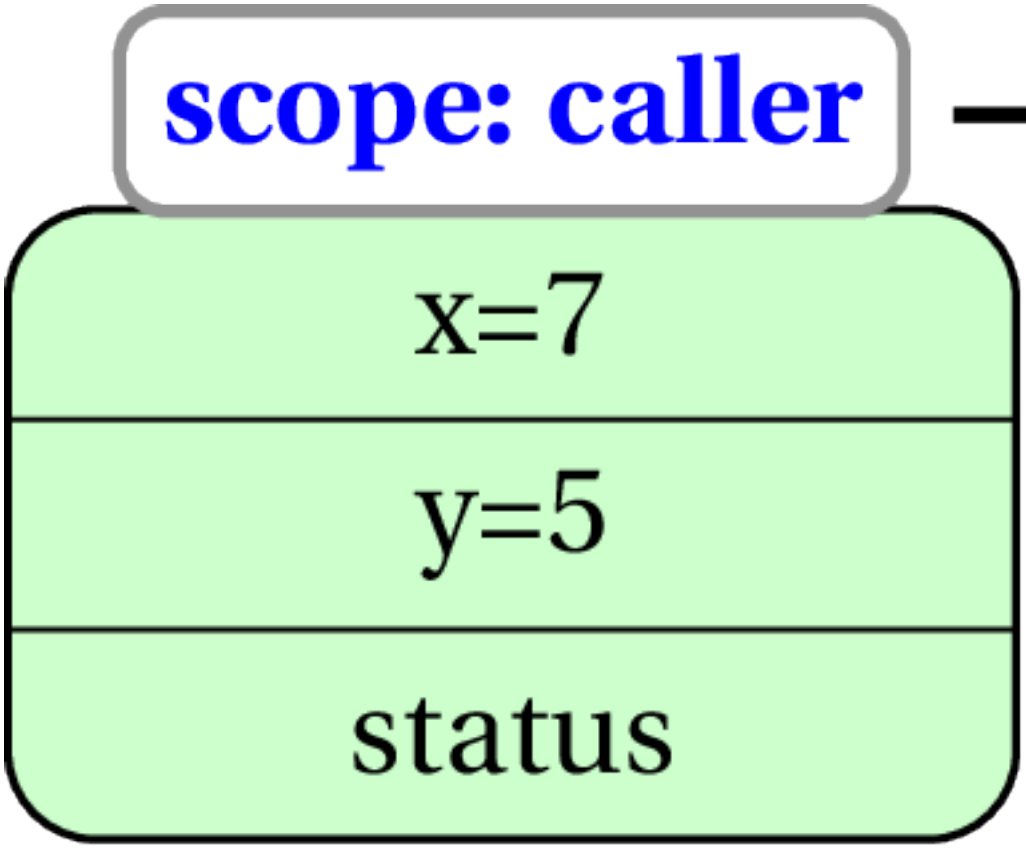
Example,

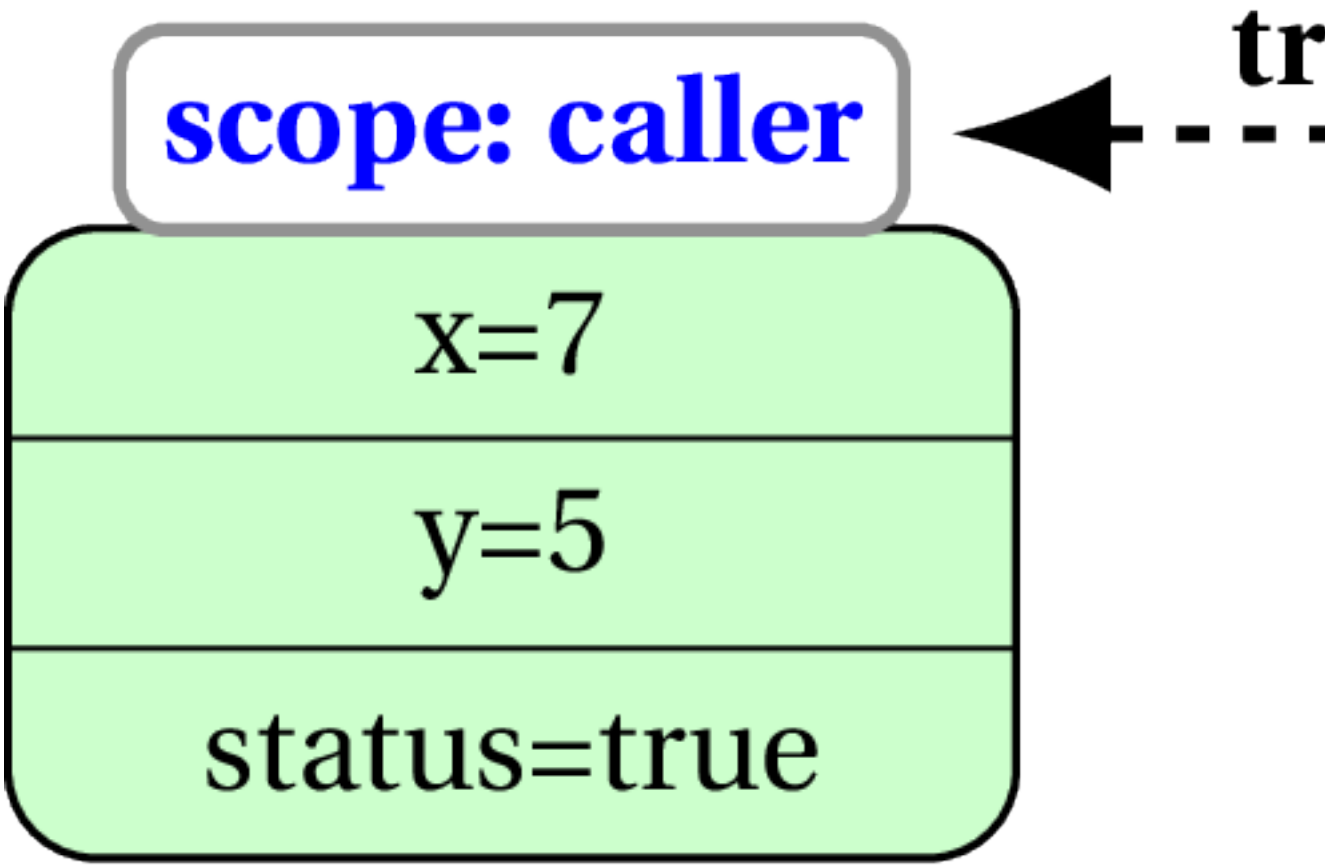
```
boolean divisible(int a, int b) {
        if(a%b == 0)
                return true;
        else
                return false;
}
```

- The function accepts two parameters, that it names **a** and **b** during the execution of the function. Here, **a, b** are called *formal parameters*.
- The function returns a value of type **boolean** back to the caller.
- Let's say the call to function **divisible** is,

```
int x = 7, y = 5;
boolean status = divisible(x+y, x-y);
```

- The integer expressions **x+y** and **x-y** are evaluated to 12 and 2 respectively. The evaluated values are known as *actual parameters* and are copied into the formal parameters **a, b** during the execution of **divisible(12, 2)** .
- Scope is transferred from the caller to the function call **divisible(12, 2)** .
- The function determines that the boolean expression **a%b == 0** is **true** , executes the if-block and returns **true** .
- The control is transferred back to the caller with the returned value **true** being copied into variable **status** .


scope: caller
x=7
y=5
status


tr
scope: caller
x=7
y=5
status=true