Rule 0.10 ⟨*Use Calldata Instead of Memory for Function Parameters*⟩

```
[...]                                        [...]
contract A {                                 contract A′ {
  [...]                                        [...]
  function f(T memory param)                   function f(T calldata param)
    external {                           =       external {
    [...]                                        [...]
    stmts                                        stmts
  }                                            }
  [...]                                        [...]
}                                            }
```

where

param is a function parameter with data location **memory** in contract A;

T is a reference type (e.g., array, struct, string, or bytes);

f is an **external** function;

stmts represents the sequence of statements in the function body.

provided

The function f has **external** visibility;

The parameter *param* is not modified within the function body (read-only access);

No memory copy of *param* is required within *stmts*;

T is a reference type that supports calldata location;

All accesses to *param* in *stmts* are compatible with calldata's read-only nature.

Invariant:

Let s_i and s'_i be the initial state of A and A', respectively.

Let s_f and s'_f be the state reached by A and A', respectively, after $A.f()$ and $A'.f()$ are executed from s_i and s'_i, respectively.

Then, the coupling invariant is

$$\forall\, s_i, s'_i \;.\; (s_i = s'_i) \rightarrow (s_f = s'_f)$$