proc sacar (inout p: MC<T> , e: T) {

requiere { e ∈ p ∧ p = p₀ }

asegura {

(p₀.data[e] = 1 ⟶ p.data = delkey(p₀.data, e)) ∧

(p₀.data[e] > 1 ⟶ p.data = setkey(p₀.data, e, p₀.data[e] - 1))

}

proc multiplicidad (in p: MC<T> , in e: T) : int {

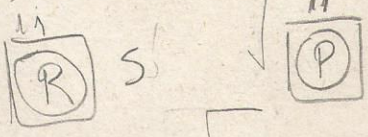
requiere { e ∈ p.data }

asegura { res = p.data[e] }

Nota También depende, pero si queremos dar la oportunidad de poner cualquier e:T

(e ∉ p.data ⟶ res = 0) ∧

(e ∈ p.data ⟶ res = p.data[e])



¿Puedo utilizar tipos de datos de esta manera para los obs?

(b)

TAD Multidict<k, seq<v>>

obs data : dict<k, seq<v>> ← puede que ocurra un pred igualdad

proc crear () : Multidict<k, Seq<v>> {

asegura { |res.data| = 0 }

}