

Algoritmos y Estructuras de Datos II

TADS: Implementaciones de contadores

18 de abril de 2018

Clase de hoy

- 1 TAD contador
 - Especificación
 - Interface
 - Implementación

Especificación del TAD Contador

module TADContador **where**

data Contador = Inicial
 | Incrementar Contador

es_inicial :: Contador → Bool

decrementar :: Contador → Contador

- - se aplica solo a un Contador que no sea Inicial

es_inicial Inicial = True

es_inicial (Incrementar c) = False

decrementar (Incrementar c) = c

Interface

type counter = ... {- no sabemos aún cómo se implementará -}

proc init (**out** c: counter) {Post: c ~ Inicial}

{Pre: c ~ C} **proc** inc (**in/out** c: counter) {Post: c ~ Incrementar C}

{Pre: c ~ C \wedge \neg is_init(c)}

proc dec (**in/out** c: counter)

{Post: c ~ decrementar C}

fun is_init (c: counter) **ret** b: **bool** {Post: b = (c ~ Inicial)}

Implementación natural

- Lo más natural es implementarlo con un natural o un entero.
- **type** counter = **nat**
- **proc** init (**out** c: counter)
 c:= 0
end proc
 {Post: c ~ Inicial}
- {Pre: c ~ C}
proc inc (**in/out** c: counter)
 c:= c+1
end proc
 {Post: c ~ Incrementar C}

Implementación natural

- $\{\text{Pre: } c \sim C \wedge \neg \text{is_init}(c)\}$
proc dec (**in/out** c: counter)
 c:= c-1
end proc
 $\{\text{Post: } c \sim \text{decrementar } C\}$
- **fun** is_init (c: counter) **ret** b: **bool**
 b:= (c = 0)
end fun
 $\{\text{Post: } b = (c \sim \text{Inicial})\}$
- Todas las operaciones son $\mathcal{O}(1)$.

Implementación rara

- Pero otras implementaciones son posibles:
- **type** counter = **int**
- **proc** init (**out** c: counter)
 c:= 17
end proc
- **proc** inc (**in/out** c: counter)
 c:= c-4
end proc

Implementación rara

- **proc** dec (**in/out** c: counter)
 c:= c+4
end proc
- **fun** is_init (c: counter) **ret** b: **bool**
 b:= (c = 17)
end fun
- Todas las operaciones son $\mathcal{O}(1)$.

Implementación rara 2

- **type** counter = nat
- **proc** init (**out** c: counter)
 c:= 1
end proc
- **proc** inc (**in/out** c: counter)
 c:= c*2
end proc
- **proc** dec (**in/out** c: counter)
 c:= c / 2
end proc

Implementación rara 2

- **fun** is_init (c: counter) **ret** b: **bool**
 b:= (c = 1)
end fun
- Todas las operaciones son $\mathcal{O}(1)$.
- ¿Cuál es el inconveniente de esta implementación?

Observaciones

- Cuando especificamos definiremos qué deben hacer las operaciones.
- Al implementar, definimos el cómo y prestamos atención a la eficiencia.
- En las 3 implementaciones que dimos del TAD Contador, todas las operaciones resultaron constantes (¡buenísimo!).
- Pero la primera fue más eficiente en el uso del espacio.