

Hola!

Algoritmos y Estructuras de Datos

Diego Bendersky, 27/8/2024

Bienvenidos!

Práctica de Algoritmos



Diego



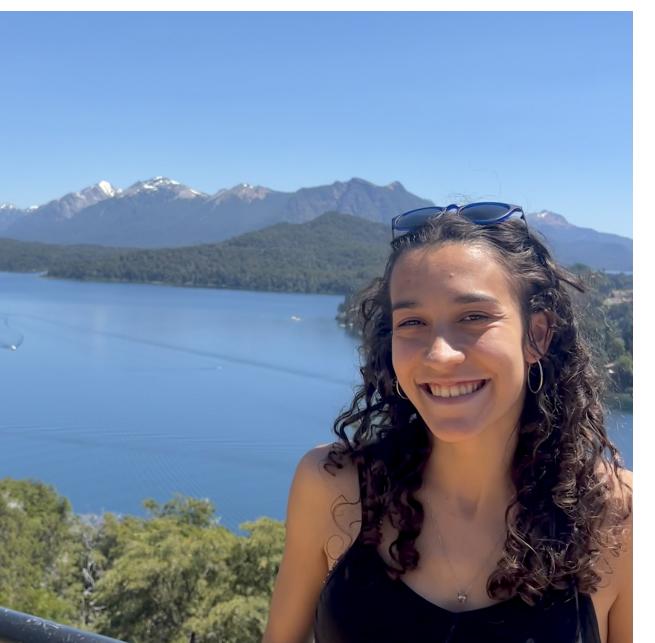
Augusto



Camilo



Ulises



Belén



Lucía



Alejandro



Valentino

Bienvenidos!

Práctica de Algoritmos

- miércoles de 9 a 14 en la Magna del 1
- vamos a repasar la teórica, resolver ejercicios y ***responder consultas***
- nos manejamos con ***guías de ejercicios*** y ***apuntes***

Aprobación

- dos parciales, cada uno con un recuperatorio (al final de la cursada)
- dos TPs grupales y talleres de laboratorio
- la materia es promocionable
 - más de 5 en cada parcial, 6 o más de promedio
 - todos los TPs y talleres aprobados

No a la violencia en Exactas

- construyamos entre todxs un ambiente libre de violencias

Seguimos en lucha

- Sueldos dignos
- Condiciones de trabajo adecuadas

Lógica y especificación

Algoritmos y Estructuras de Datos

Diego Bendersky, 27/8/2024

Lógica

Qué vamos a ver hoy?

- Repaso de lógica proposicional
- Tautologías y contradicciones
- Relaciones de fuerza
- Lógica trivaluada
- Cuantificadores

Lógica proposicional

- variables

- p, q, r, \dots

- símbolos

- \wedge \vee \neg \rightarrow \leftrightarrow

- fórmulas

- $(p \wedge (q \vee r)) \leftrightarrow ((p \wedge q) \vee (p \wedge r))$ ✓

- $(p) \wedge \wedge (q \vee)) \leftrightarrow \rightarrow ((p \ q)$ ✗

Valuaciones

Ejercicio 1. Determinar los valores de verdad de las siguientes proposiciones cuando el valor de verdad de a , b y c es *verdadero* y el de x e y es *falso*.

a) $(\neg x \vee b)$

b) $((c \vee (y \wedge a)) \vee b)$

c) $\neg(c \vee y)$

d) $\neg(y \vee c)$

e) $(\neg(c \vee y) \leftrightarrow (\neg c \wedge \neg y))$

f) $((c \vee y) \wedge (a \vee b))$

g) $((c \vee y) \wedge (a \vee b)) \leftrightarrow (c \vee (y \wedge a) \vee b)$

h) $(\neg c \wedge \neg y)$

Tautologías y contradicciones

Ejercicio 4. Determinar si las siguientes fórmulas son tautologías, contradicciones o contingencias.

a) $(p \vee \neg p)$

b) $(p \wedge \neg p)$

c) $((\neg p \vee q) \leftrightarrow (p \rightarrow q))$

d) $((p \wedge q) \rightarrow p)$

e) $((p \wedge (q \vee r)) \leftrightarrow ((p \wedge q) \vee (p \wedge r)))$

f) $((p \rightarrow (q \rightarrow r)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow r)))$

Relación de fuerza

- p es ***mas fuerte*** que q si:
 - siempre que se cumple p, se cumple q
 - formalmente: $p \rightarrow q$ es una tautología
 - las cosas que hacen cumplir q *incluyen* a las que hacen cumplir p

Relación de fuerza

Ejercicio 5. Dadas las proposiciones lógicas α y β , se dice que α es más fuerte que β si y sólo si $\alpha \rightarrow \beta$ es una tautología. En este caso, también decimos que β es más débil que α . Determinar la relación de fuerza de los siguientes pares de fórmulas:

a) True, False

b) $(p \wedge q), (p \vee q)$

c) $p, (p \wedge q)$

d) $p, (p \vee q)$

e) p, q

f) $p, (p \rightarrow q)$

¿Cuál es la proposición más fuerte y cuál la más débil de las que aparecen en este ejercicio?

Lógica trivaluada

- Tres valores de verdad
 - Verdadero, Falso, Indefinido
 - Operadores "luego"
 - \wedge_L \vee_L \rightarrow_L

$$P(n) = \sqrt{n} > 2 \quad \text{X}$$

$$P(n) = n \geq 0 \wedge_L \sqrt{n} > 2 \quad \checkmark$$

Cuantificadores

- Dado un predicado $P(x : \mathbb{Z})$ (por ejemplo, " $P(x) = x$ es par")
- $(\forall x : \mathbb{Z})(P(x))$
 - es equivalente a: $\cdots \wedge P(-2) \wedge P(-1) \wedge P(0) \wedge P(1) \wedge P(2) \wedge \cdots$
- $(\exists x : \mathbb{Z})(P(x))$
 - es equivalente a: $\cdots \vee P(-2) \vee P(-1) \vee P(0) \vee P(1) \vee P(2) \vee \cdots$

Cuantificadores

- Ejemplos:
 - $P = (\forall x : \mathbb{Z})(x \neq 0 \rightarrow (x < 0 \vee x > 0))$
 - $P(y) = (\forall x : \mathbb{Z})(x \cdot y \bmod 2 = 0)$
 - $P(z) = (\exists x : \mathbb{Z})(x > z)$

Cuantificadores

Ejercicio 9. Sea $P(x : \mathbb{Z})$ y $Q(x : \mathbb{Z})$ dos predicados cualquiera. Explicar cuál es el error de traducción a fórmulas de los siguientes enunciados. Dar un ejemplo en el cuál sucede el problema y luego corregirlo.

a) “Todos los naturales menores a 10 cumplen P ”

$$(\forall i : \mathbb{Z})((0 \leq i < 10) \wedge P(i))$$

b) “Algún natural menor a 10 cumple P ”

$$(\exists i : \mathbb{Z})((0 \leq i < 10) \rightarrow P(i))$$

c) “Todos los naturales menores a 10 que cumplen P , cumplen Q ”:

$$(\forall x : \mathbb{Z})((0 \leq x < 10) \rightarrow (P(x) \wedge Q(x)))$$

d) “No hay ningún natural menor a 10 que cumpla P y Q ”:

$$\neg((\exists x : \mathbb{Z})(0 \leq x < 10 \wedge P(x))) \wedge \neg((\exists x : \mathbb{Z})(0 \leq x < 10 \wedge Q(x)))$$

Cuantificadores

Ejercicio 10. Sean $P(x : \mathbb{Z})$ y $Q(x : \mathbb{Z})$ dos predicados cualesquiera que nunca se indefinen. Escribir el predicado asociado a cada uno de los siguientes enunciados:

- “Existe un único número natural menor a 10 que cumple P ”
- “Existen al menos dos números naturales menores a 10 que cumplen P ”
- “Existen exactamente dos números naturales menores a 10 que cumplen P ”
- “Todos los enteros pares que cumplen P , no cumplen Q ”
- “Si un entero cumple P y es impar, no cumple Q ”
- “Todos los enteros pares cumplen P , y todos los enteros impares que no cumplen P cumplen Q ”

Relación de fuerza

Ejercicio 12. Sean $P(x : \mathbb{Z})$ y $Q(x : \mathbb{Z})$ dos predicados cualesquiera que nunca se indefinen y sean a, b y k enteros. Decidir en cada caso la relación de fuerza entre las dos fórmulas:

a) $P(3)$

$$(\forall n : \mathbb{Z})((0 \leq n < 5) \rightarrow P(n))$$

b) $P(3)$

$$(\exists n : \mathbb{Z})(0 \leq n < 5 \wedge P(n))$$

c) $(\forall n : \mathbb{Z})((0 \leq n < 10 \wedge P(n)) \rightarrow Q(n))$

$$(\forall n : \mathbb{Z})((0 \leq n < 10) \rightarrow Q(n))$$

d) $(\exists n : \mathbb{Z})(0 \leq n < 10 \wedge P(n) \wedge Q(n))$

$$(\forall n : \mathbb{Z})((0 \leq n < 10) \rightarrow Q(n))$$

e) $k = 0 \wedge (\exists n : \mathbb{Z})(0 \leq n < 10 \wedge P(n) \wedge Q(n))$

$$k = 0 \wedge ((\forall n : \mathbb{Z})((0 \leq n < k) \rightarrow Q(n)))$$

Recreo!

Algoritmos y Estructuras de Datos

Diego Bendersky, 27/8/2024

Especificación

Qué vamos a ver hoy?

- Escribir e interpretar predicados lógicos
- Especificar problemas (procs)
 - Pasaje de parámetros (in/inout)
 - Requiere y asegura
 - Descomposición en auxiliares (pred y aux)

Predicados lógicos

- Es una manera de darle nombre a una fórmula
- Recibe *parámetros*

```
pred esDivisiblePor (n:  $\mathbb{Z}$ , m:  $\mathbb{Z}$ ) {  
    n  $\text{mód}$  m = 0  
}
```

Tipos de parámetros

Ver apunte!

245
entero

'b'
char

$\langle 2,3,4 \rangle$
secuencia

3.14
real

'diego'
string

$(Diego,2)$
tupla

true
bool

$(x : 245, y : 635)$
registro

Secuencias

Operación	Sintaxis
secuencia por extensión	$\langle \rangle, \langle x, y, z \rangle$
longitud	$ s , length(s), s.length$
pertenece	$i \in s$
indexación	$s[i]$
cabeza	$head(s)$
cola	$tail(s)$
concatenación	$concat(s_1, s_2), s_1 + + s_2$
subsecuencia	$subseq(s, i, j)$
cambiar elemento	$setAt(s, i, val)$

- Todos los elementos tienen el mismo tipo. Ej: $seq < \mathbb{Z} >$

$\langle 1,2,3 \rangle$ ✓

$\langle \rangle$ ✓

$\langle 1, Diego, 3 \rangle$ ✗

Predicados lógicos

Ejercicio 2. Escriba los siguientes predicados sobre números enteros en lenguaje de especificación:

- a) pred *sonCoprimos* ($x, y : \mathbb{Z}$) que sea verdadero si y sólo si x e y son coprimos.

- b) pred *mayorPrimoQueDivide* ($x : \mathbb{Z}, y : \mathbb{Z}$) que sea verdadero si y es el mayor primo que divide a x .

Predicados lógicos

Ejercicio 4. Escriba los siguientes predicados auxiliares sobre secuencias de enteros, aclarando los tipos de los parámetros que recibe:

- a) *esPrefijo*, que determina si una secuencia es prefijo de otra.
- b) *estáOrdenada*, que determina si la secuencia está ordenada de menor a mayor.
- c) *hayUnoParQueDivideAlResto*, que determina si hay un elemento par en la secuencia que divide a todos los otros elementos de la secuencia.
- d) *enTresPartes*, que determina si en la secuencia aparecen (de izquierda a derecha) primero 0s, después 1s y por último 2s. Por ejemplo $\langle 0, 0, 1, 1, 1, 1, 2 \rangle$ cumple con *enTresPartes*, pero $\langle 0, 1, 3, 0 \rangle$ o $\langle 0, 0, 0, 1, 1 \rangle$ no. ¿Cómo modificaría la expresión para que se admitan cero apariciones de 0s, 1s y 2s (es decir, para que por ejemplo $\langle 0, 0, 0, 1, 1 \rangle$ o $\langle \rangle$ sí cumplan *enTresPartes*)?

Funciones auxiliares

- Similar a los predicados pero devuelven algo de algún tipo

,

$$\text{aux } \text{suma10}(a : \mathbb{Z}) : \mathbb{Z} =$$
$$a + 10$$

,

$$\text{aux } \text{sumaSeq}(s : \text{seq}(\mathbb{Z})) : \mathbb{Z} =$$
$$\sum_{i=0}^{|s|-1} s[i]$$

IfThenElse

- Es una función auxiliar
- IfThenElse(cond, x, y)
 - Cond es expresión de tipo bool (o un predicado)
 - x, y son expresiones *del mismo tipo*
 - devuelve algo del mismo tipo que x e y
- **No se usa para expresar causalidad (“si pasa P entonces pasa Q”)**

IfThenElse

- Ejemplos:
- Cantidad de apariciones del elemento a en la secuencia s

$$\begin{aligned} \text{aux } \text{apariciones}(s : \text{seq}\langle\mathbb{Z}\rangle, a : \mathbb{Z}) : \mathbb{Z} = \\ \sum_{i=0}^{|s|-1} \text{IfThenElse}(s[i] = a, 1, 0) \end{aligned}$$

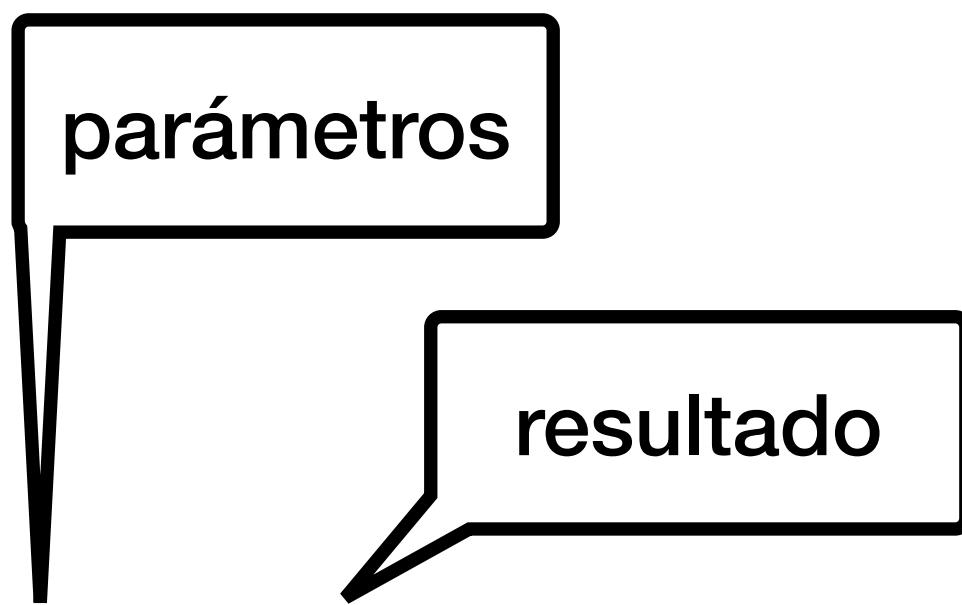
- Suma de todos los elementos pares de la secuencia s

$$\begin{aligned} \text{aux } \text{sumaDePares}(s : \text{seq}\langle\mathbb{Z}\rangle, a : \mathbb{Z}) : \mathbb{Z} = \\ \sum_{i=0}^{|s|-1} \text{IfThenElse}(\text{mód } 2 = 0, s[i], 0) \end{aligned}$$

Especificación de problemas

- Vamos a definir el **qué** y no el **cómo**
- Vamos a usar lógica para los requiere y asegura

Especificación de problemas



```
proc raizCuadrada (in n: Z) : Z
    requiere {n ≥ 0}
    asegura {res ≥ 0 ∧ res × res = n}
```

- Al resultado lo llamamos *res* (sólo en el asegura!)

Pasaje de parámetros

- **in**: no se modifica. A la salida vale lo mismo que a la entrada
- **inout**: se puede modificar. El valor a la salida puede ser diferente al valor de entrada

```
proc sumame (inout a: Z, in b: Z)
```

```
    requiere {a = A0}
```

```
    asegura {a = A0 + b}
```

si es inout, aviso que voy a llamar A_0
al valor que tenía al entrar

uso el valor inicial en la
salida

en el asegura, "a" se
refiere al valor de salida

Descomposición en auxiliares

- para hacer más clara la lectura, los requiere y asegura se pueden descomponer en predicados y funciones auxiliares

```
proc ordenar (inout a: seq<Z>)
    requiere {a = A0}
    asegura {
        tienenLosMismosElementos(a, A0) ∧
        estaOrdenada(a)
    }
    pred tienenLosMismosElementos (a: seq<Z>, b: seq<Z>) {
        ...
    }
    pred estaOrdenada (a: seq<Z>) {
        ...
    }
```

Especificación de problemas

2.4. Especificación de problemas

Ejercicio 12. Especificar los siguientes problemas:

- a) Dado un entero, decidir si es par
- b) Dado un entero n y otro m , decidir si n es un múltiplo de m
- c) Dado un entero, listar todos sus divisores positivos (sin duplicados)
- d) Dado un entero positivo, obtener su descomposición en factores primos. Devolver una secuencia de tuplas (p, e) , donde p es un factor primo y e es su exponente, ordenada en forma creciente con respecto a p

Especificación de problemas

Ejercicio 17. Especificar los siguientes problemas de modificación de secuencias:

- a) proc primosHermanos(inout $l : seq\langle \mathbb{Z} \rangle$), que dada una secuencia de enteros mayores a dos, reemplaza dichos valores por el número primo menor más cercano. Por ejemplo, si $l = \langle 6, 5, 9, 14 \rangle$, luego de aplicar $primosHermanos(l)$, $l = \langle 5, 3, 7, 13 \rangle$
- b) proc reemplazar(inout $l : seq\langle Char \rangle$, in $a, b : Char$), que reemplaza todas las apariciones de a en l por b
- c) proc limpiarDuplicados(inout $l : seq\langle Char \rangle$) : $seq\langle Char \rangle$, que elimina los elementos duplicados de l dejando sólo su primera aparición (en el orden original). Devuelve además una secuencia con todas las apariciones eliminadas (en cualquier orden)

Especificación de problemas

Ejercicio 18. Especificar los siguientes problemas. En todos los casos es recomendable ayudarse escribiendo predicados y funciones auxiliares.

- a) Se desea especificar el problema `reemplazarNúmerosPerfectos`, que dada una secuencia de enteros devuelve la secuencia pero con los valores que se corresponden con números perfectos reemplazados por el índice donde se encuentran. Se llama números perfectos a aquellos naturales mayores a cero que son iguales a la suma de sus divisores positivos propios (divisores incluyendo al 1 y sin incluir al propio número). Por ejemplo, $\text{reemplazarNúmerosPerfectos}([0, 3, 9, 6, 4, 28, 7]) = [0, 3, 9, 3, 4, 5, 7]$, donde los únicos números reemplazados son el 6 y el 28 porque son los únicos números perfectos de la secuencia.

Chau!

Algoritmos y Estructuras de Datos

Diego Bendersky, 27/8/2024