

Corso di Programmazione

I Prova di accertamento del 22 Gennaio 2018 / A

cognome e nome

Selezione degli esercizi proposti.

3. Programmazione in Scheme

Dato un intero $n \geq 0$, la procedura `powers-of-two` restituisce la lista delle potenze di due distinte che lo compongono, la cui somma è n , ordinate in ordine decrescente. In altri termini, ciascuna delle potenze di due rappresentate nella lista corrisponde a uno dei bit '1' della notazione binaria di n , a partire da quello più significativo. Esempi:

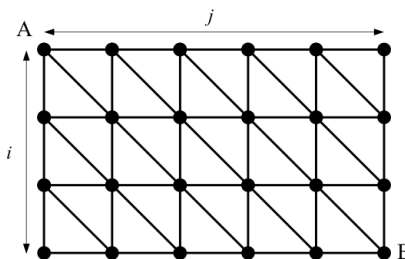
```
(powers-of-two 0) → ()
(powers-of-two 1) → (1)
(powers-of-two 5) → (4 1)
(powers-of-two 8) → (8)
(powers-of-two 26) → (16 8 2)
(powers-of-two 45) → (32 8 4 1)
```

Definisci un programma in Scheme per realizzare la procedura `powers-of-two`.

```
(define powers-of-two ;restituisce la stringa binaria
  (lambda (n) ;n: INTERO >= 0
    (cond ( (= n 0) ;CASO BASE
            null
          )
          ( (= n 1)
            (list 1)
          )
          (else
            (let ( (lg (inexact->exact
                        (floor (/ (log n) (log 2)) )
                      )
                  ) ;Log2(n) : approssimato difetto INTERO
              )
              (cons (expt 2 lg) ;(potenza di 2) < n più vicina a n stesso
                    (powers-of-two (- n (expt 2 lg))) ; n - (potenza di 2)
                  )
            )
          )
    )
  )
)
```

4. Ricorsione ad albero

Completa la definizione della procedura `manhattan-var`, progettata per risolvere una variante del “problema di Manhattan” in cui i nodi sono connessi, oltre che dai consueti tratti orizzontali e verticali, anche da tratti diagonali che scendono verso destra come illustrato nella figura a fianco. Interessa conoscere in quanti modi diversi ci si può spostare dal nodo A al nodo B lungo un percorso di lunghezza minima che attraversi *esattamente* k tratti diagonali (e non di più: osserva che gli spostamenti in diagonale abbreviano i percorsi rispetto al caso di spostamenti orizzontali a destra e verticali in basso, ma non è consentito utilizzarne più di k ; se $k = 0$, in particolare, ci si riconduce alla soluzione del problema originale). Esempi:



```
(manhattan-var 3 2 0) → 10   (manhattan-var 3 2 2) → 3   (manhattan-var 2 2 2) → 1
```

```
(define manhattan-var ; val: intero
  (lambda (i j k) ; i, j, k: interi non negativi tali che k ≤ i e k ≤ j
    (let (
      (x (if (= i k) 0 (manhattan-var (- i 1) j k)))
      (y (if _____ 0 (manhattan-var i (- j 1) k)))
      (z (if (= k 0) 0 (manhattan-var _____ )))
    )
      (if (and (> i 0) (> j 0))
        (+ _____ )
        _____
      )
    )
  )
)
```

5. Verifica formale della correttezza

In relazione alla procedura `ufo`, riportata qui a fianco, si può dimostrare che per ogni intero $k \geq 0$:

$$(ufo \ 7 \cdot 2^k) \rightarrow 3 \cdot 2^{k+1} + 1$$

Dimostra questa proprietà per induzione sui valori di k attenendoti allo schema delineato qui sotto.

```
(define ufo
  (lambda (n)
    (cond ((= n 1)
          1)
          ((even? n)
           (- (* 2 (ufo (quotient n 2))) 1))
          (else
           (+ (* 2 (ufo (quotient n 2))) 1))
          )))
```

- Formalizza la proprietà generale da dimostrare:

$$P(k): (ufo \ 7 \cdot 2^k) = 3 \cdot 2^{k+1} + 1 \text{ per qualsiasi } k \geq 0$$

- Formalizza la proprietà che esprime il caso / i casi base:

$$P(0): (ufo \ 7 \cdot 2^0) = (ufo \ 7 \cdot 1)$$

Si potrebbe anche dimostrare $P(1)$ in modo da essere ancora più sicuri che la proprietà valga per ogni $n \geq 0$

- Formalizza l'ipotesi induttiva:

$$P(n): (ufo \ 7 \cdot 2^n) = 3 \cdot 2^{n+1} + 1$$

- Formalizza la proprietà da dimostrare come passo induttivo:

$$P(n+1): (ufo \ 7 \cdot 2^{n+1}) ==? 3 \cdot 2^{n+2} + 1$$

Il passo induttivo incrementa di 1 la variabile di riferimento(N) al risultato dell'ipotesi induttiva

- Dimostra il caso / i casi base:

$$\begin{aligned} (ufo \ 7) &\rightarrow (even? \ 7) = false \rightarrow (ufo \ 3) \rightarrow (even? \ 3) = false \rightarrow 2 \cdot (ufo \ 1) - 1 = 1 \\ 2 \cdot 1 + 1 &= 3 == (ufo \ 3) \rightarrow (ufo \ 7) = 2 \cdot (ufo \ 3) + 1 = 2 \cdot 3 + 1 = 7 \\ P(0) : \text{CASO BASE} &= (ufo \ 7) = 7 \end{aligned}$$

- Dimostra il passo induttivo:

$$\begin{aligned} (ufo \ 7 \cdot 2^{n+1}) &= (even? \ 7 \cdot 2^{n+1}) = true \text{ per ogni } n \geq 0 \rightarrow (- (* 2 (ufo (quotient n 2))) 1)) \\ (quotient \ 7 \cdot 2^{n+1} \ 2) &= 7 \cdot 2^n \cdot 2^{-1} = 7 \cdot 2^{n-1} = 7 \cdot 2^n \\ (ufo \ 7 \cdot 2^n) &= \text{secondo la proprietà } P(n) = 3 \cdot 2^{n+1} + 1 \end{aligned}$$

$$\begin{aligned} (- (* 2 (ufo \ 7 \cdot 2^n)) 1) &= [2 \cdot (3 \cdot 2^{n+1} + 1)] - 1 = 3 \cdot 2^{n+2} + 2 - 1 = \\ &= 3 \cdot 2^{n+2} + 1 = \text{PRINCIPIO DI INDUZIONE CONFERMATO} \end{aligned}$$

Corso di Programmazione

I Prova di accertamento del 22 Gennaio 2018 / B

cognome e nome

Selezione degli esercizi proposti.

3. Programmazione in Scheme

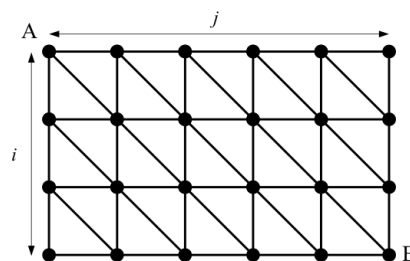
Dato un intero $n \geq 0$, la procedura `powers-of-two` restituisce la lista delle potenze di due distinte che lo compongono, la cui somma è n , ordinate in ordine crescente. In altri termini, ciascuna delle potenze di due rappresentate nella lista corrisponde a uno dei bit '1' della notazione binaria di n , a partire dal quello meno significativo. Esempi:

<code>(powers-of-two 0)</code>	\rightarrow	<code>()</code>	<code>(powers-of-two 16)</code>	\rightarrow	<code>(16)</code>
<code>(powers-of-two 1)</code>	\rightarrow	<code>(1)</code>	<code>(powers-of-two 22)</code>	\rightarrow	<code>(2 4 16)</code>
<code>(powers-of-two 9)</code>	\rightarrow	<code>(1 8)</code>	<code>(powers-of-two 53)</code>	\rightarrow	<code>(1 4 16 32)</code>

Definisci un programma in Scheme per realizzare la procedura `powers-of-two`.

4. Ricorsione ad albero

Completa la definizione della procedura `manhattan-var`, progettata per risolvere una variante del “problema di Manhattan” in cui i nodi sono connessi, oltre che dai consueti tratti orizzontali e verticali, anche da tratti diagonali che scendono verso destra come illustrato nella figura a fianco. Interessa conoscere in quanti modi diversi ci si può spostare dal nodo A al nodo B lungo un percorso di lunghezza minima che attraversi *esattamente* k tratti diagonali (e non di più: osserva che gli spostamenti in diagonale abbreviano i percorsi rispetto al caso di spostamenti orizzontali a destra e verticali in basso, ma non è consentito utilizzarne più di k ; se $k = 0$, in particolare, ci si riconduce alla soluzione del problema originale). Esempi:



`(manhattan-var 3 2 0) \rightarrow 10` `(manhattan-var 3 2 2) \rightarrow 3` `(manhattan-var 2 2 2) \rightarrow 1`

```
(define manhattan-var      ; val: intero
  (lambda (i j k)         ; i, j, k: interi non negativi tali che  $k \leq i$  e  $k \leq j$ 
    (if (or (= i 0) (= j 0))

        .....

        (let (
          (x (if (> i k) (manhattan-var (- i 1) j k) 0))

          (y (if ..... (manhattan-var i (- j 1) k) 0))

          (z (if (> k 0) (manhattan-var ..... ) 0))
        )

          (+ ..... )

        ))
```

5. Verifica formale della correttezza

In relazione alla procedura `ufo`, riportata qui a fianco, si può dimostrare che per ogni intero $k \geq 0$:

$$(ufo\ 5 \cdot 2^k) \rightarrow 2^{k+1} + 1$$

Dimostra questa proprietà per induzione sui valori di k attenendoti allo schema delineato qui sotto.

```
(define ufo
  (lambda (n)
    (cond ((= n 1)
          1)
          ((even? n)
           (- (* 2 (ufo (quotient n 2))) 1))
          (else
           (+ (* 2 (ufo (quotient n 2))) 1))
          )))
```

- Formalizza la proprietà generale da dimostrare:
- Formalizza la proprietà che esprime il caso / i casi base:
- Formalizza l'ipotesi induttiva:
- Formalizza la proprietà da dimostrare come passo induttivo:
- Dimostra il caso / i casi base:
- Dimostra il passo induttivo: