

Corso di Programmazione

I Prova di accertamento del 22 Gennaio 2018 / A

cognome e nome

Selezione degli esercizi proposti.

3. Programmazione in Scheme

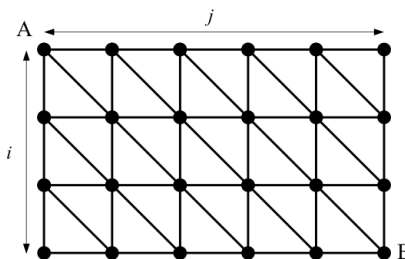
Dato un intero $n \geq 0$, la procedura `powers-of-two` restituisce la lista delle potenze di due distinte che lo compongono, la cui somma è n , ordinate in ordine decrescente. In altri termini, ciascuna delle potenze di due rappresentate nella lista corrisponde a uno dei bit '1' della notazione binaria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

<code>(powers-of-two 0)</code>	\rightarrow	<code>()</code>	<code>(powers-of-two 8)</code>	\rightarrow	<code>(8)</code>
<code>(powers-of-two 1)</code>	\rightarrow	<code>(1)</code>	<code>(powers-of-two 26)</code>	\rightarrow	<code>(16 8 2)</code>
<code>(powers-of-two 5)</code>	\rightarrow	<code>(4 1)</code>	<code>(powers-of-two 45)</code>	\rightarrow	<code>(32 8 4 1)</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) → 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 $k \geq 0$:

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

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:
- 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:

Corso di Programmazione

I Prova di accertamento del 22 Gennaio 2018 / B

cognome e nome

Selezione degli esercizi proposti.

3. Programmazione in Scheme

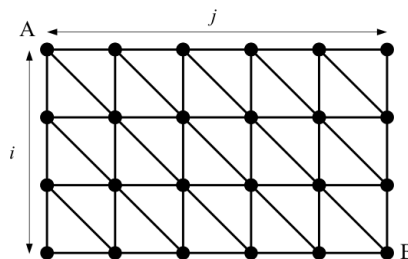
Dato un intero $n \geq 0$, la procedura `powers-of-two` restituisce la lista delle potenze di due distinte che lo compongono, la cui somma è n , ordinate in ordine crescente. In altri termini, ciascuna delle potenze di due rappresentate nella lista corrisponde a uno dei bit '1' della notazione binaria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