Ricorsione generale e ricorsione di coda a confronto

Esempio di ricorsione generale:

Algoritmo di moltiplicazione “alla russa”

domanda :	m, n	m, n	m, n	m = ___ , n = ___
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 5 , n = 25
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 5 , n = 25
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 5 , n = 25
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 10 , 12
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 10 , n = 12
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 10 , n = 12
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 10 , n = 12
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 20 , 6
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 20 , n = 6
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 20 , n = 6
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 20 , n = 6
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 40 , 3
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 40 , n = 3
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 40 , n = 3
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 40 , n = 3
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 80 , 1
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 80 , n = 1
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 80 , n = 1
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 80 , n = 1
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 160 , 0
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 160 , n = 0
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 160 , n = 0
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 160 , n = 0
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	—
risposta :	0	↓ ??	↓ m + ??	↓ 0

domanda :	m, n	m, n	m, n	m = 80 , n = 1
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 160 , 0 ... → 0
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 80 , n = 1
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 160 , 0 ... → 0
risposta :	0	↓ ??	↓ m + ??	↓ 80 + 0 = 80

domanda :	m, n	m, n	m, n	m = 40 , n = 3
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 80 , 1 ... → 80
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 40 , n = 3
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 80 , 1 ... → 80
risposta :	0	↓ ??	↓ m + ??	↓ 40 + 80 = 120

domanda :	m, n	m, n	m, n	m = 20 , n = 6
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 40 , 3 ... → 120
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 20 , n = 6
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 40 , 3 ... → 120
risposta :	0	↓ ??	↓ m + ??	↓ 120

domanda :	m, n	m, n	m, n	m = 10 , n = 12
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 20 , 6 ... → 120
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 10 , n = 12
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 20 , 6 ... → 120
risposta :	0	↓ ??	↓ m + ??	↓ 120

domanda :	m, n	m, n	m, n	m = 5 , n = 25
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 10 , 12 ... → 120
risposta :	0	↓ ??	↓ m + ??	

domanda :	m, n	m, n	m, n	m = 5 , n = 25
(define mul ; val: intero (lambda (m n) ; m, n: interi non negativi (cond ((= n 0) 0) ((even? n) (mul (* 2 m) (quotient n 2))) (else (+ m (mul (* 2 m) (quotient n 2)))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2*m, n/2 ... → ??	delega per 2*m, n/2 ... → ??	delega per 10 , 12 ... → 120
risposta :	0	↓ ??	↓ m + ??	↓ 5 + 120 = 125

Esempio di ricorsione di coda:

Moltiplicazione “del contadino russo”

domanda :	m, n, p	m, n, p	m, n, p	m = ___ , n = ___ , p = ___
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 5 , n = 25 , p = 0
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 5 , n = 25 , p = 0
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 5 , n = 25 , p = 0
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 10 , 12 , 5
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 10 , n = 12 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 10 , n = 12 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 10 , n = 12 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 20 , 6 , 5
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 20 , n = 6 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 20 , n = 6 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 20 , n = 6 , p = ___
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 40 , 3 , 5
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 40 , n = 3 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 40 , n = 3 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 40 , n = 3 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 80 , 1 , 45
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 80 , n = 1 , p = 45
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 80 , n = 1 , p = 45
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 80 , n = 1 , p = 45
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 160 , 0 , 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 160 , n = 0 , p = 125
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 160 , n = 0 , p = 125
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 160 , n = 0 , p = 125
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	—
risposta :	p	↓ ??	↓ ??	↓ 125

domanda :	m, n, p	m, n, p	m, n, p	m = 80 , n = 1 , p = 45
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 160 , 0 , 125 ... → 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 80 , n = 1 , p = 45
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 160 , 0 , 125 ... → 125
risposta :	p	↓ ??	↓ ??	↓ 125

domanda :	m, n, p	m, n, p	m, n, p	m = 40 , n = 3 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 80 , 1 , 45 ... → 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 40 , n = 3 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 80 , 1 , 45 ... → 125
risposta :	p	↓ ??	↓ ??	↓ 125

domanda :	m, n, p	m, n, p	m, n, p	m = 20 , n = 6 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 40 , 3 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 20 , n = 6 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 40 , 3 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	↓ 125

domanda :	m, n, p	m, n, p	m, n, p	m = 10 , n = 12 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 20 , 6 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 10 , n = 12 , p = 5
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false true
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 20 , 6 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	↓ 125

domanda :	m, n, p	m, n, p	m, n, p	m = 5 , n = 25 , p = 0
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 10 , 12 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	

domanda :	m, n, p	m, n, p	m, n, p	m = 5 , n = 25 , p = 0
(define mul ; val: intero (lambda (m n p) ; m, n, p: interi non negativi (cond ((= n 0) p) ((even? n) (mul (* 2 m) (quotient n 2) p)) (else (mul (* 2 m) (quotient n 2) (+ m p))))))	n = 0	n pari	altrimenti (n dispari)	false false Ok
delega :	—	delega per 2m, n/2, p ... → ??	delega per 2m, n/2, m+p ... → ??	delega per 10 , 12 , 5 ... → 125
risposta :	p	↓ ??	↓ ??	↓ 125

Esempio di ricorsione di coda:

Algoritmo di Euclide per il massimo comun divisore

domanda :	x, y	x, y	x, y	x = ___ , y = ___
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 60 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 60 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 60 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 42 , 18
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 42 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 42 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 42 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 24 , 18
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 24 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 24 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 24 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 18
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 12
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 12
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 12
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 12
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 6
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 6
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 6
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	—
risposta :	x	↓ ??	↓ ??	↓ 6

domanda :	x, y	x, y	x, y	x = 6 , y = 12
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 6 ... → 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 12
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 6 ... → 6
risposta :	x	↓ ??	↓ ??	↓ 6

domanda :	x, y	x, y	x, y	x = 6 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 12 ... → 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 6 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false true
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 12 ... → 6
risposta :	x	↓ ??	↓ ??	↓ 6

domanda :	x, y	x, y	x, y	x = 24 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y x < y altrimenti (x > y)			false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 24 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 6 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	↓ 6

domanda :	x, y	x, y	x, y	x = 42 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y x < y altrimenti (x > y)			false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 24 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 42 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 24 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	↓ 6

domanda :	x, y	x, y	x, y	x = 60 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y x < y altrimenti (x > y)			false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 42 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	

domanda :	x, y	x, y	x, y	x = 60 , y = 18
(define gcd ; val: intero (lambda (x y) ; x, y: interi positivi (cond ((= x y) x) ((< x y) (gcd x (– y x))) (else (gcd (– x y) y)))))	x = y	x < y	altrimenti (x > y)	false false Ok
delega :	—	delega per x, y–x ... → ??	delega per x–y, y ... → ??	delega per 42 , 18 ... → 6
risposta :	x	↓ ??	↓ ??	↓ 6