35.Write a function even num(my−list) which returns the list of even numbers

from a list. Do not use the predicate even?. Exemple:

> ( even num ’ ( 1 2 3 4 4 5 7 8 ) )

( 2 4 4 8 )

Solution:

(define (elem-even ls)

(define res '()) ; definim lista de rezultat

(cond [(list? ls)

(for ([i ls]) ; iteram prin lista primita ca argument

(cond ((eq? (remainder i 2) 0) ;cand numarl e par il adaugam la lista de rezultat

(set! res (cons i res))

)))

(reverse res)]

[else "Argumentul nu este lista!"]))

(elem-even '(1 2 2 2 3 4 5 6))

(elem-even 67)

12. Write a tail recursive program which calculates the factorial of a natural number.

Solution:

(define (factorial n)

(if (number? n) ;Am verificat daca argumentul primit e numar

(if (= n 0) 1 ; cazul de baza

(\* n (factorial (- n 1)))

)

(quote "argumentul nu e numar"))

)

(factorial 6) ;720

11. Write a function shift right (List1) such that the returned list is List1 ”as

a rotational shift” with one element to the right. Example:

> ( shift\_right ’( 1 2 3 4 5 ) )

( 5 1 2 3 4 )

> ( shift\_right ’ ( 5 1 2 3 4 ) )

( 4 5 1 2 3 )

Solution:

(define (shift-right lst)

(if (list? lst) ;Am verificat daca argumentul primit e lista

(if (null? (cdr lst)) ; Am verificar daca lista e vida

lst

(cons (car (reverse lst)) (reverse (cdr (reverse lst)))))

(quote "argumentul nu e lista")))

(shift-right '(1 2 3 4 5))