a)

remove\_by\_n(l1,l2,…,lm, n, k) =

{ nil , daca m = 0

{ remove\_by\_n(l2,…lm, n, n) , daca k = 1

{ l1 U remove(l2,…lm, n, k-1) , daca k != 1

remove\_by\_n(L, n) = { remove\_by\_name(L, n, n)

b)

valley\_no(l1,…ln) =

{ 0 , daca n = 0

{ 0 , daca l2 = []

{ 2 , daca l3 = [] si l1 >= l2

{ 0 , daca l3 = [] si l1 < l2

{ 2 , daca l1 = l2 ( 5 5 2 3 – nu e vale)

{ 1 + valley\_no(l2,…ln) , daca l1 > l2 < l3

{ valley\_no(l2,…ln) , altfel

is\_valley(L) =

{ format nil “E vale” , daca valley\_no(L) = 1

{ format nil “NU e vale” , daca valley\_no(L) != 1

c)

extract\_numbers(l1,...ln, acc) =

{ acc , daca n = 0

{ l1 U acc , daca l1 este atom numeric

{ extract\_numbers(l2,...ln, extract\_numbers(l1, acc))

minim(a, b) =

{ a , daca a < b

{ b , altfel

minim\_list(l1,...ln) =

{ nil , n = 0

{ l1 , n = 1

{ minim(l1, minim\_list(l2,...ln)) , altfel

minim\_list\_main(L) = minim\_list(extract\_numbers(L, []))

d)

maxim(a, b) =

{ a , daca a > b

{ b , altfel

maxim\_list(l1,...ln) =

{ nil ,daca n = 0

{ l1 ,daca n = 1

{ maxim(l1, maxim\_list(l2,...ln)) , altfel

remove\_elem(l1,...l2, e) =

{ nil , daca n = 0

{ remove\_elem(l2,...ln, e) , daca l1 = e

{ l1 U remove\_elem(l2,...ln, e) , altfel

remove\_maxim(L) = remove\_elem(L, maxim\_list(extract\_numbers(L,[]))))