Ali Ghaleb

10/17/2021

COSC 341

Assignment 2::Pattern style

1. Function name: **d2b(n)**

Function description: The d2b function converts an integer into a binary number recursively.

CODE:

fun d(0) = 0

| d(n) = n mod 2 + 10 \* d(n div 2);

fun intl(L) = if L div 10 = 0 then L::nil

else L mod 10 :: intl(L div 10);

fun charl(nil) = nil

| charl(l::ls) = chr(l + 48) :: charl(ls);

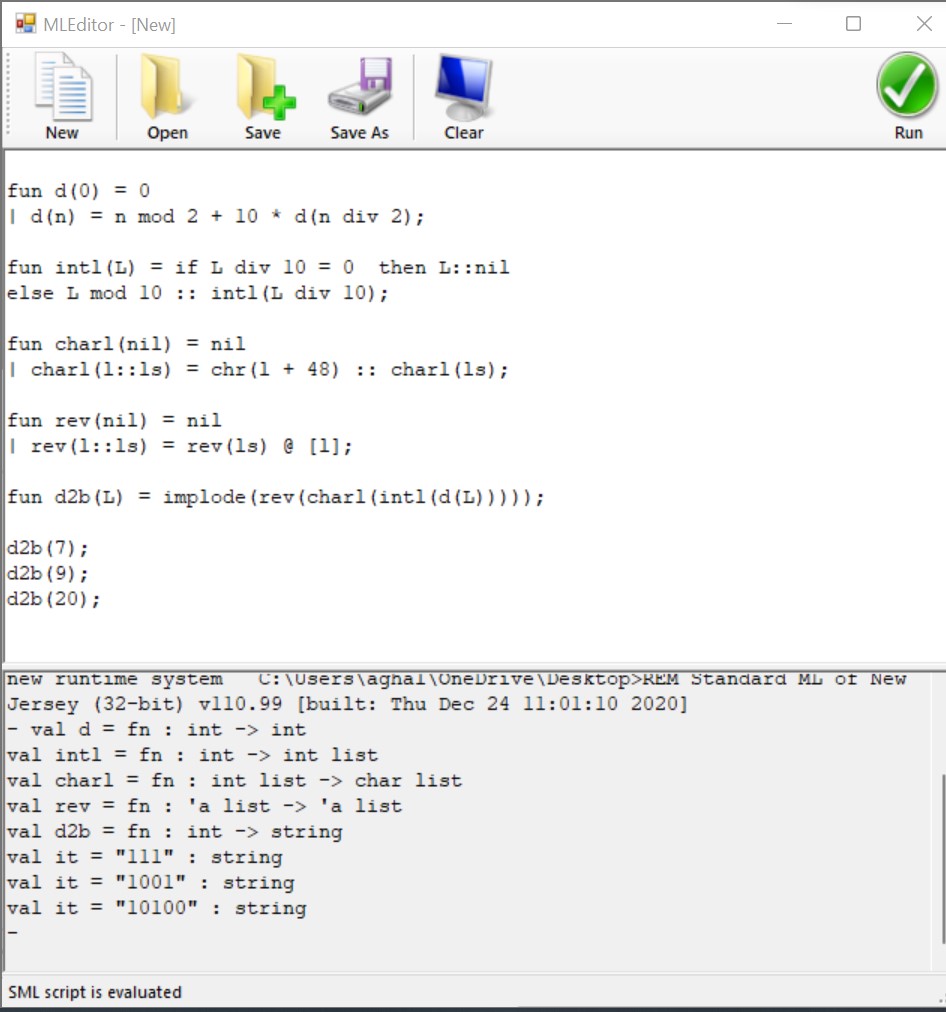
fun rev(nil) = nil

| rev(l::ls) = rev(ls) @ [l];

fun d2b(L) = implode(rev(charl(intl(d(L)))));

d2b(7);

SAMPLE RUN:



1. Function name: inde(n,L);

Function description: function inde returns the index (start from 1) of the occurrence of a given value n.

CODE:

fun rea(n,nil,index) = nil

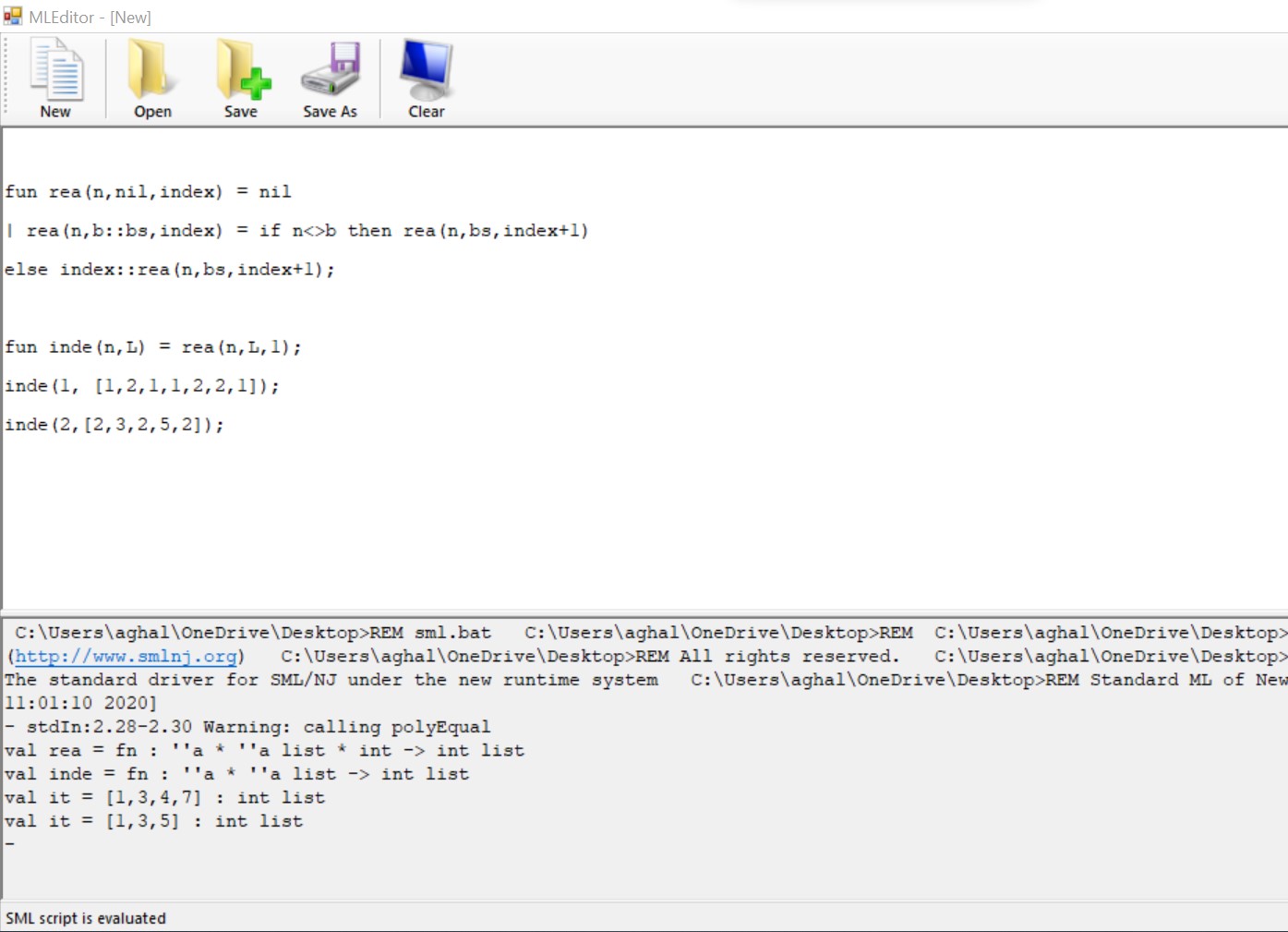
| rea(n,b::bs,index) = if n<>b then rea(n,bs,index+1)

else index::rea(n,bs,index+1);

fun inde(n,L) = rea(n,L,1);

inde(1,[1,2,1,1,2,2,1]);

SAMPLE RUN:



1. Function name: **nele(L,n);**

Function description: Repeats each element in a list. Use helper function. Two functions only.

CODE:

fun repeat(nil,n,index) = nil

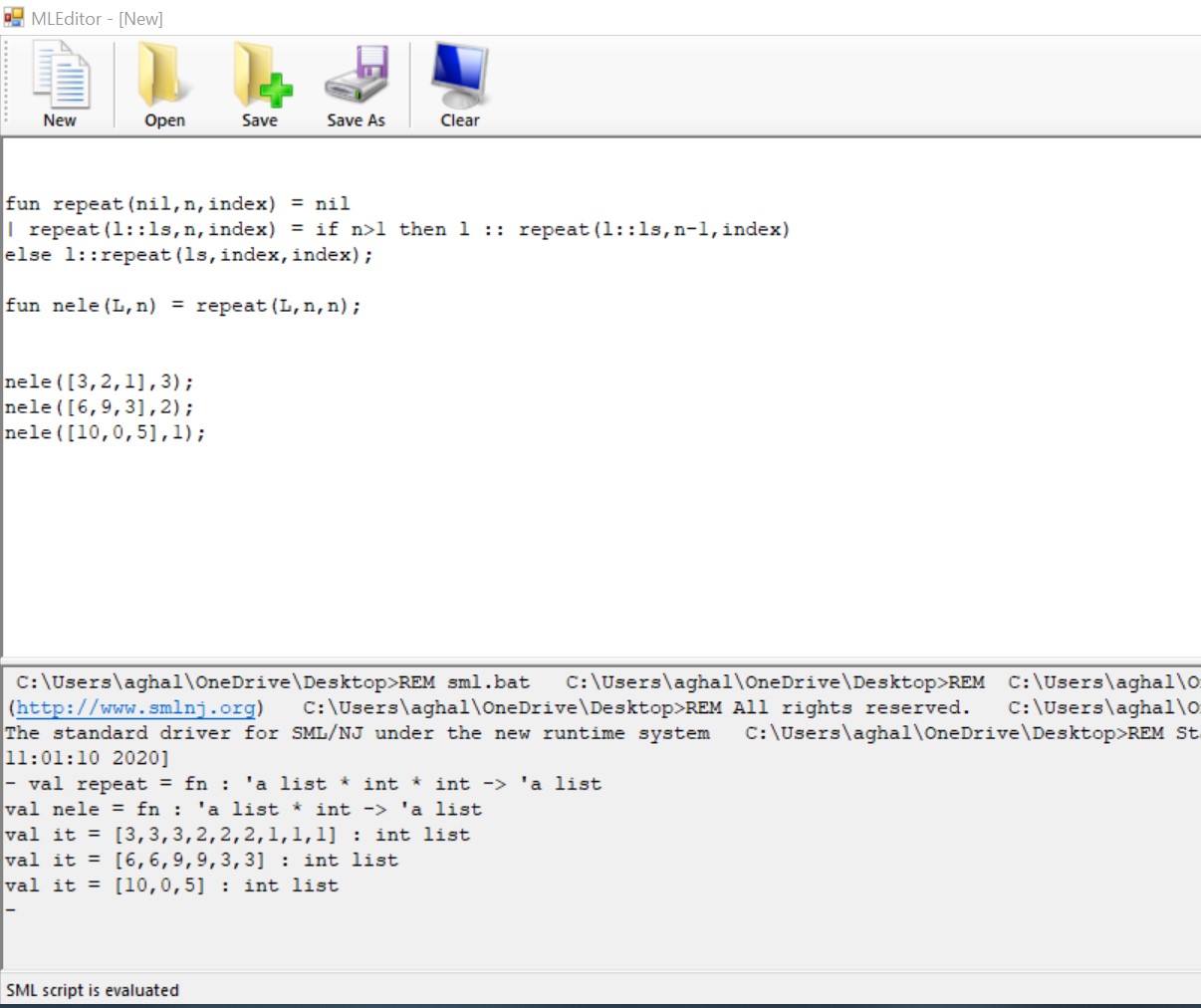
| repeat(l::ls,n,index) = if n>1 then l :: repeat(l::ls,n-1,index)

else l::repeat(ls,index,index);

fun nele(L,n) = repeat(L,n,n);

nele([3,2,1],3);

SAMPLE RUN:



1. Function name: isfact(n)

Function description: isfact(n) is a function that determines if a positive integer is a factorial number or not without formulas (two functions).

CODE:

Fun fact(n,x,l) = if n=x then true

else if x>n then false

else fact(n,x\*l,l+1);

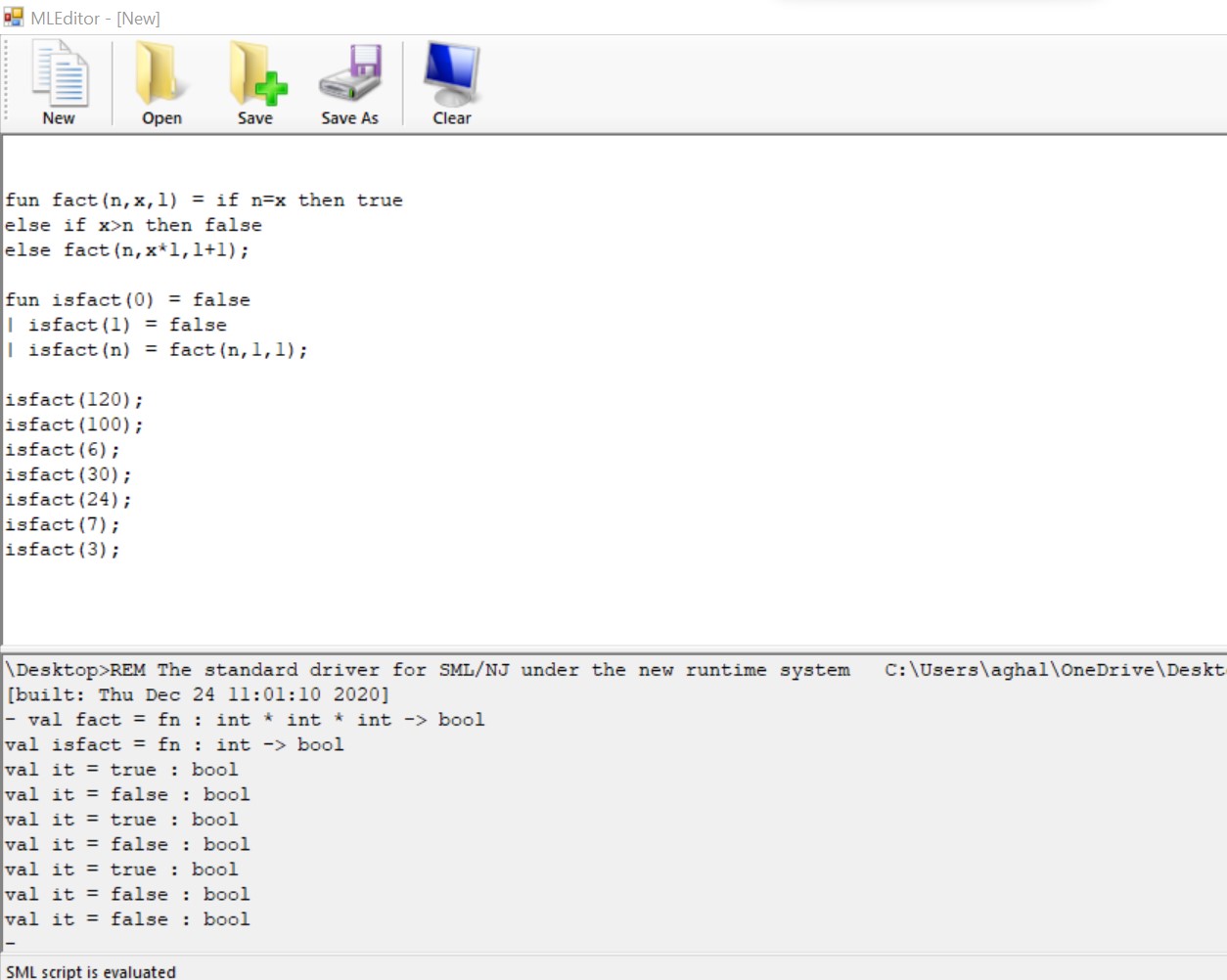
fun isfact(0) = false

| isfact(1) = false

| isfact(n) = fact(n,1,1);

isfact(120);

Sample run:



1. Function name: ntri(n)

Function description: This function generates a list of n triangular numbers.

CODE:

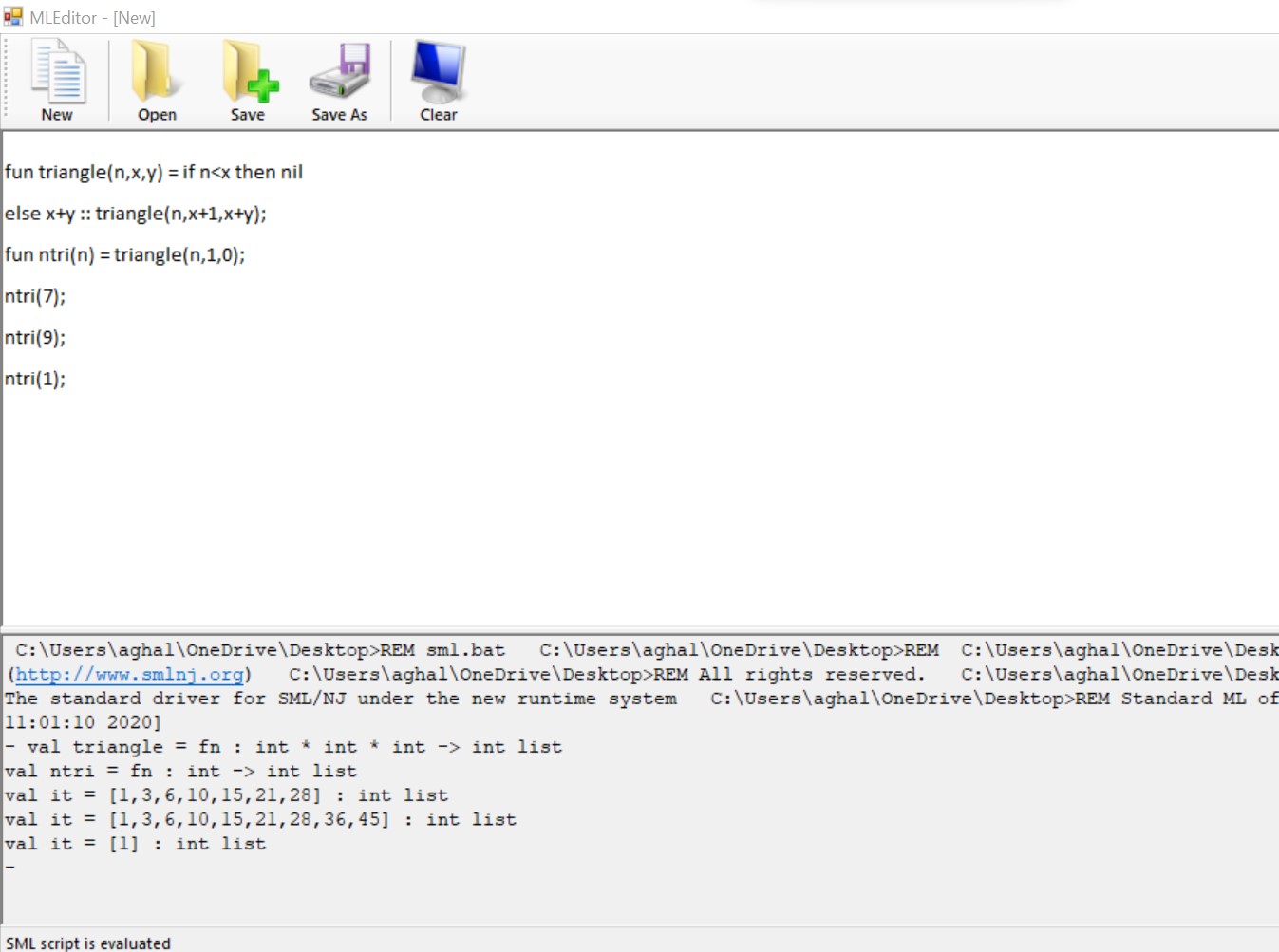
fun triangle(n,x,y) = if n<x then nil

else x+y :: triangle(n,x+1,x+y);

fun ntri(n) = triangle(n,1,0);

ntri(7);

SAMPLE RUN:



1. Function name: occr(L);

Function description: display the occurrence of an element of a lst or nil if empty in a tuple(element,occurrence)

CODE:

fun count(nil,n,list,(y,z)) = (nil,n, list, (y,z))

| count(x::xs,n,list, (y,z)) = if n = x then count(xs,n,list,(n,z+1))

else count(xs,n,x::list,(n,z));

fun occr(nil) = nil

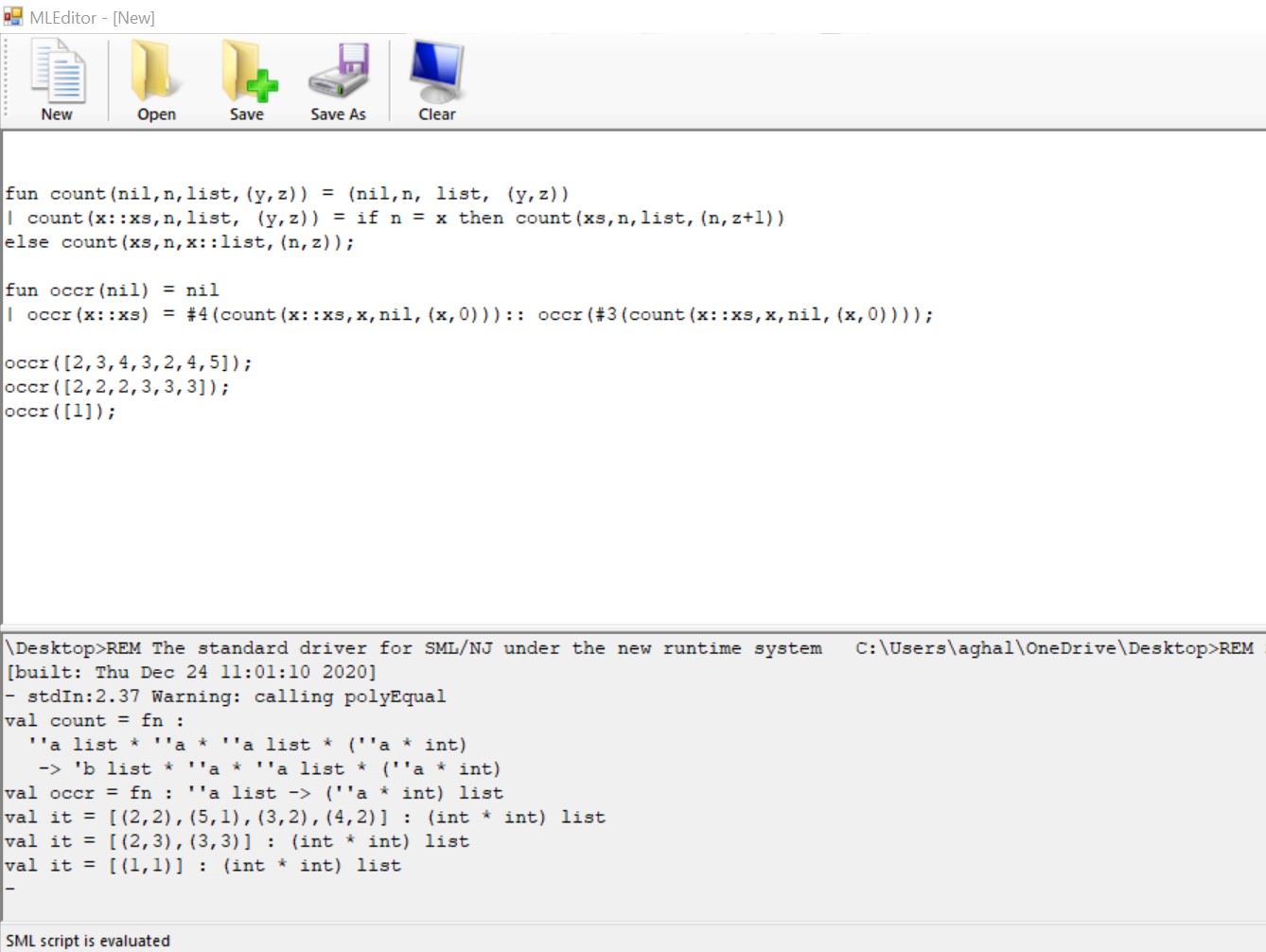
| occr(x::xs) = #4(count(x::xs,x,nil,(x,0))):: occr(#3(count(x::xs,x,nil,(x,0))));

occr([2,3,4,3,2,4,5]);

occr([2,2,2,3,3,3]);

occr([1]);

SAMPLE CODE:



1. Function name: chcase(s)

Function description: A high order function that converts input string to uppercase.

CODE:

fun simpleMap(F,nil) = nil

| simpleMap(F,l::ls) = F l:: simpleMap(F, ls);

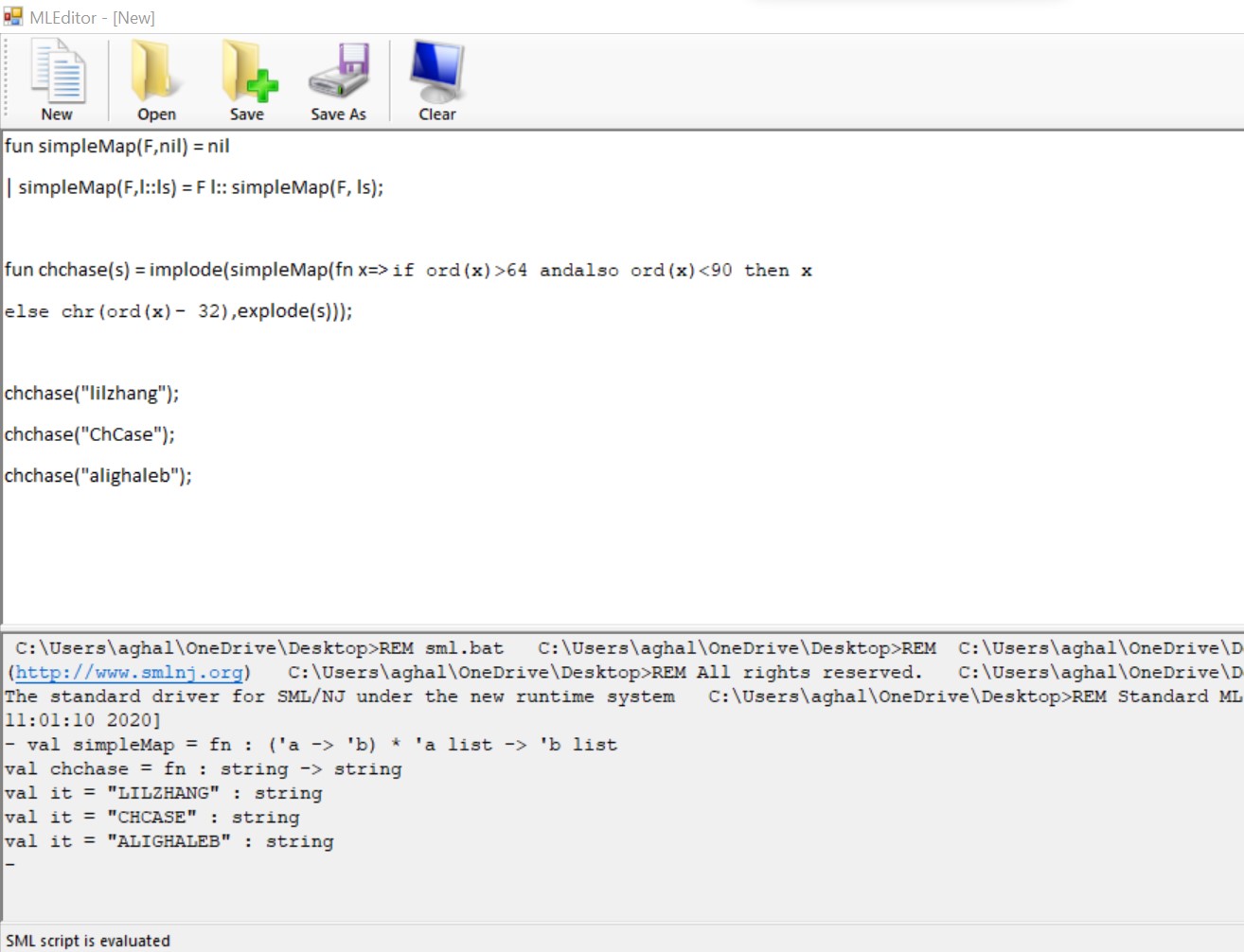
fun chchase(s) = implode(simpleMap(fn x=> if ord(x)>64 andalso ord(x)<90 then x else chr(ord(x)- 32),explode(s)));

chchase("lilzhang");

chchase("ChCase");

chchase("alighaleb");

Sample run:



8. Function name: infront1(a,L)

Function description: A high order function that inserts an element as the head of each element of a list.

CODE:

val L = [[1,2],nil,[3]];

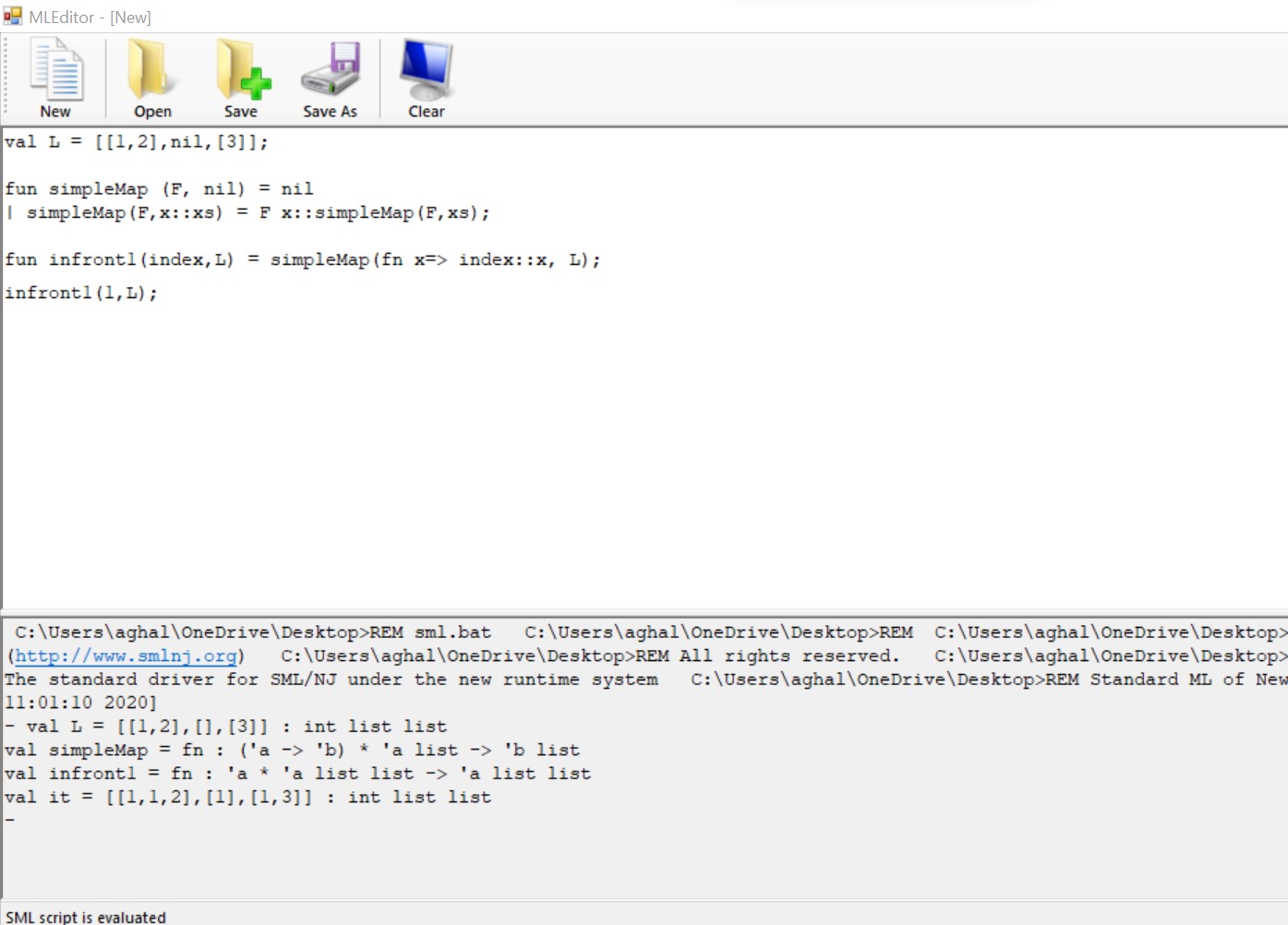
fun simpleMap (F, nil) = nil

| simpleMap(F,x::xs) = F x::simpleMap(F,xs);

fun infront1(index,L) = simpleMap(fn x=> index::x, L);

infront1(1,L);

Sample run:



1. Function name: infront(a,L)

Function description: A function that inserts an element as the head of each element of a list without using high order function.

CODE:

fun isfront2(n,nil) = nil

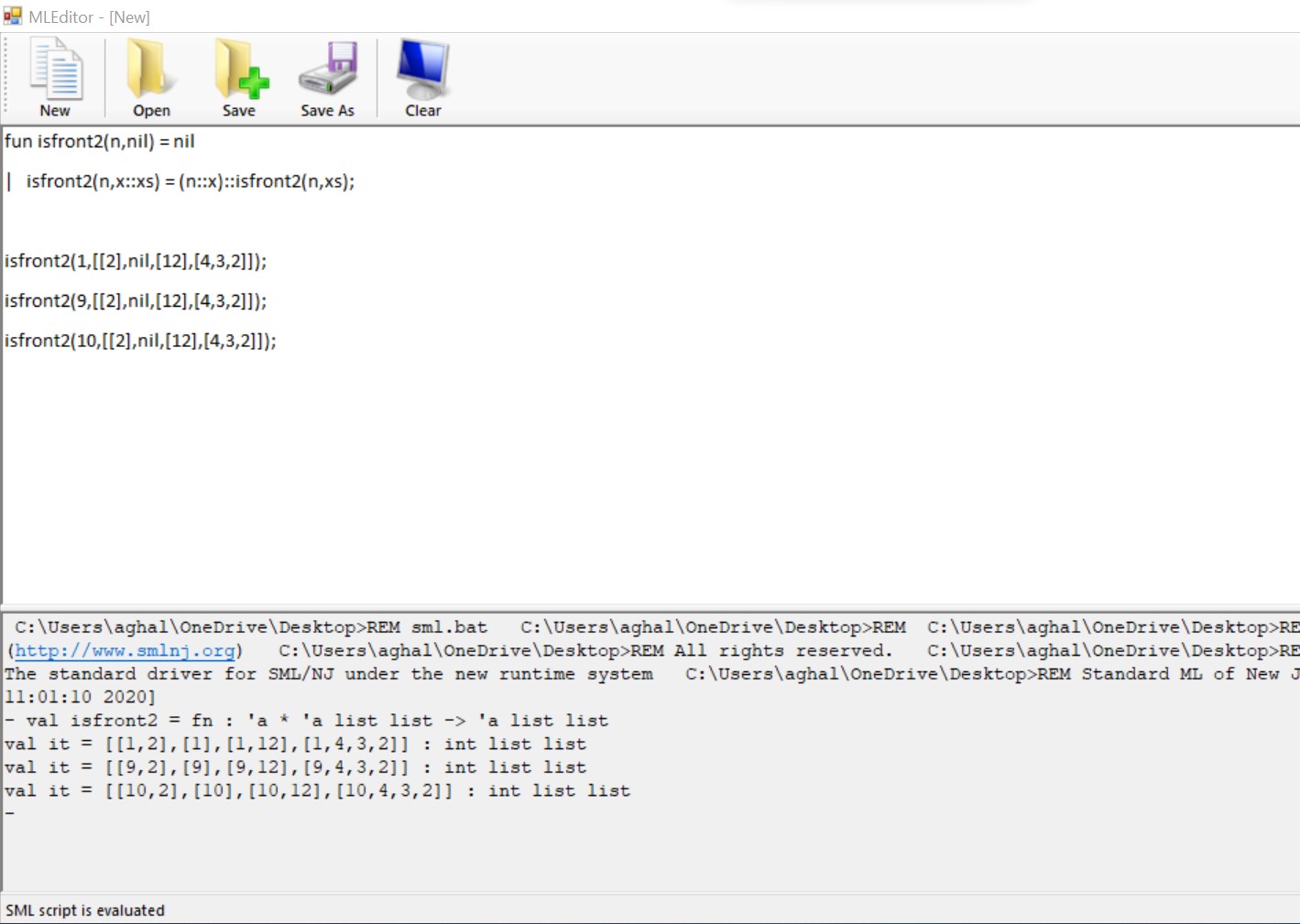
| isfront2(n,x::xs) = (n::x)::isfront2(n,xs);

isfront2(1,[[2],nil,[12],[4,3,2]]);

isfront2(9,[[2],nil,[12],[4,3,2]]);

isfront2(10,[[2],nil,[12],[4,3,2]]);

Sample run:



1. Function name: insea(n,L);

Function description: Insert an element to each position of a list.

CODE:

fun append(nil,L2) = L2

|append(L1::L1s,L2) = L1 :: append(L1s,L2);

fun insertElement(L1,a,nil) = append(L1,[a]) :: nil

| insertElement(nil,a,L2::L2s) = append([a],L2::L2s) :: insertElement([L2],a,L2s)

| insertElement(L1,a,L2::L2s) = append(append(L1,[a]), L2::L2s) :: insertElement(append(L1,[L2]),a,L2s);

fun insea(n, list) = insertElement(nil,n,list);

insea(4,[1,2,3]);

SAMPLE RUN:

