Ali Ghaleb

10/17/2021

COSC 341

Assignment 2: ifelse style

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

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

CODE:

fun d(n) = if n=0 then 0

else 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(L) = if null L then nil

else chr(hd L + 48) :: charl(tl L);

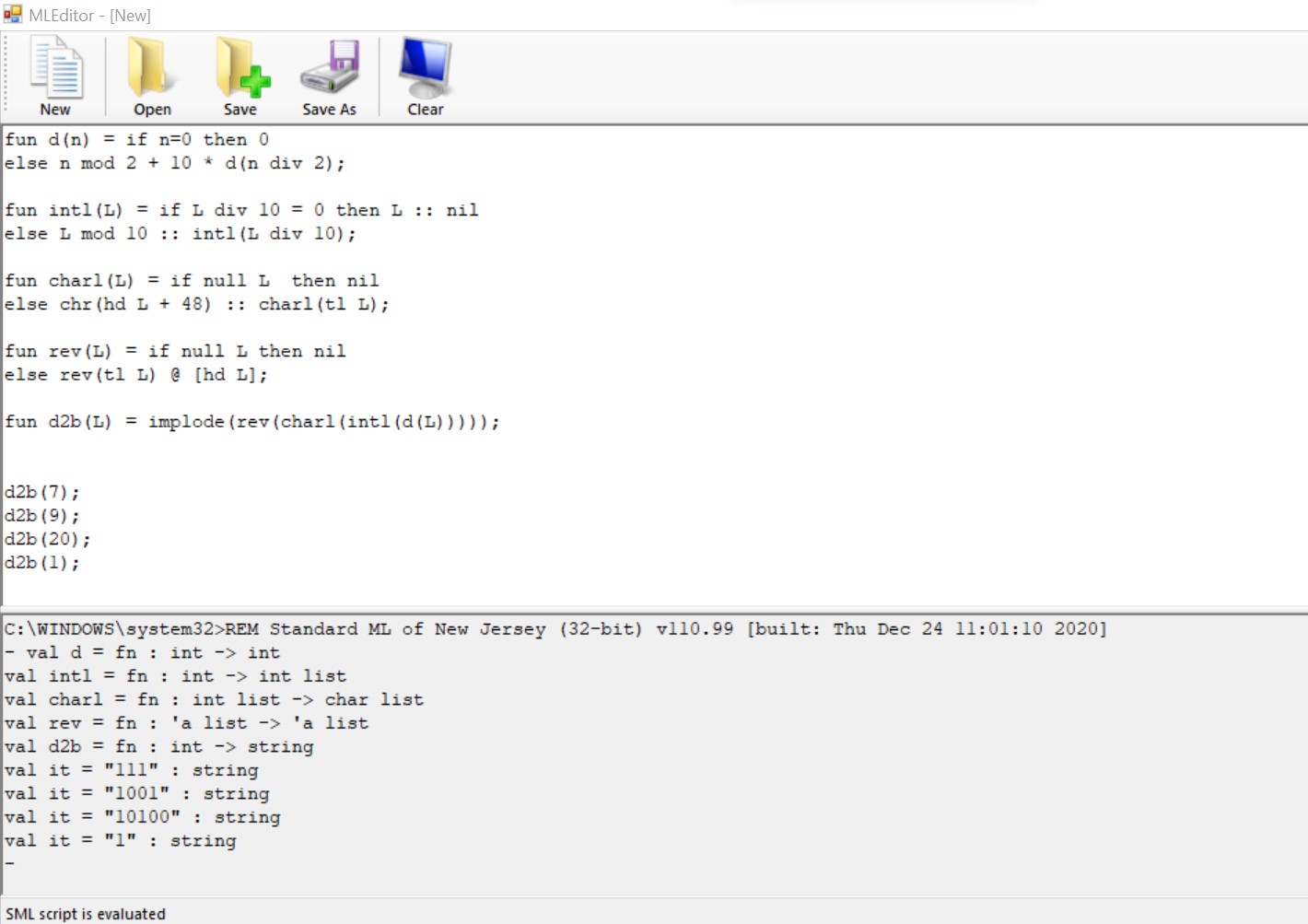
fun rev(L) = if null L then nil

else rev(tl L) @ [hd 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)

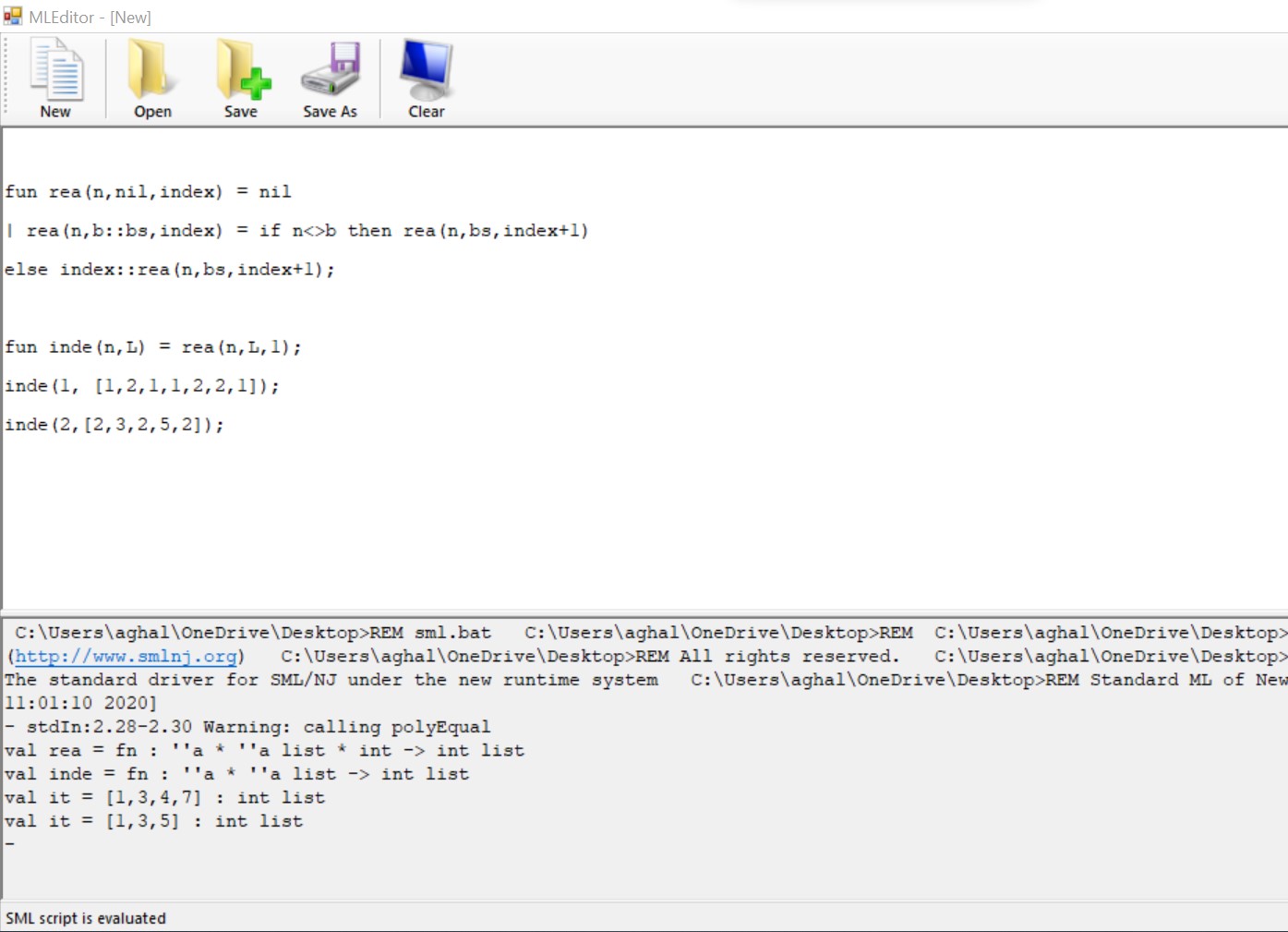
else index::rea(n,bs,index+1);

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

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

inde(2,[2,3,2,5,2]);

SAMPLE RUN:



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

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

CODE:

fun neleR(L,n,indexreset) = if null L then nil

else if n>1 then hd L :: neleR(L,n-1,indexreset)

else hd L :: neleR(tl L,indexreset,indexreset);

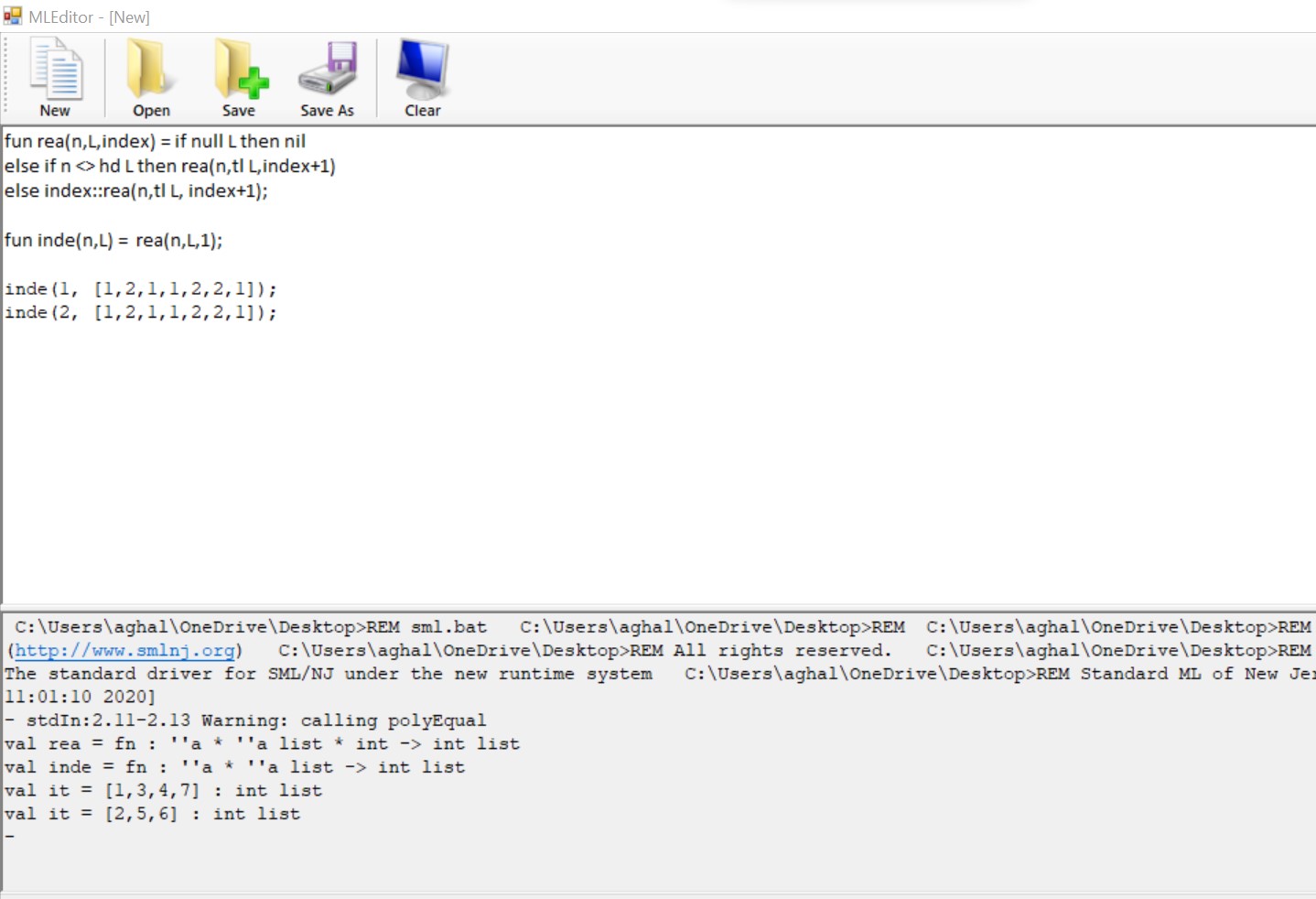
fun nele(L,n) = neleR(L,n,n);

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

nele([6,9,3],2);

nele([10,0,5],1);

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);

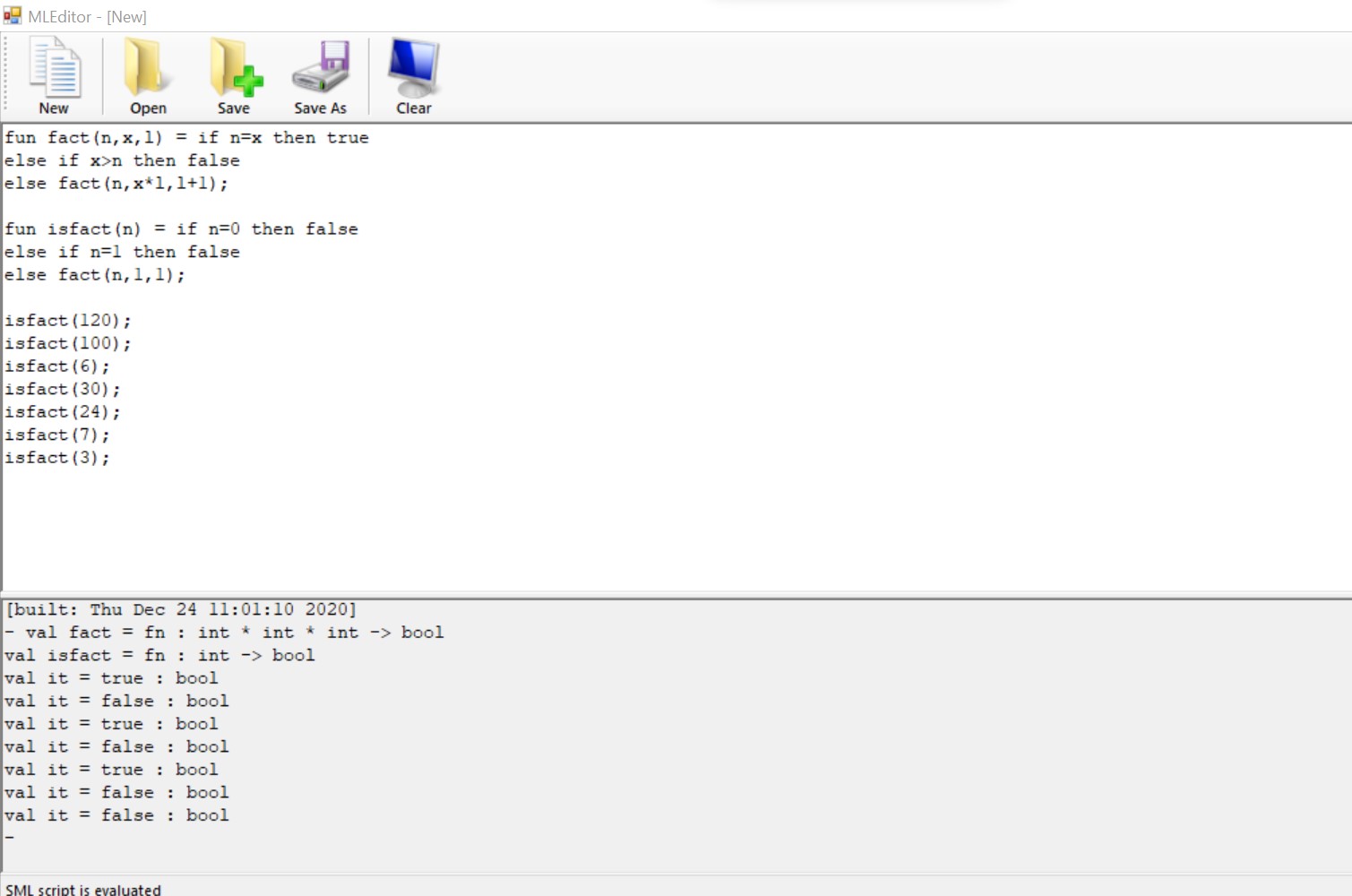
fun isfact(n) = if n=0 then false

else if n=1 then false

else 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);

ntri(9);

ntri(1);

SAMPLE RUN:

Graphical user interface, text, application, email

Description automatically generated

1. Function name: chcase(s)

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

Code:

fun simpleMap(F,L) = if null L then nil

else F(hd L) :: simpleMap(F, tl L);

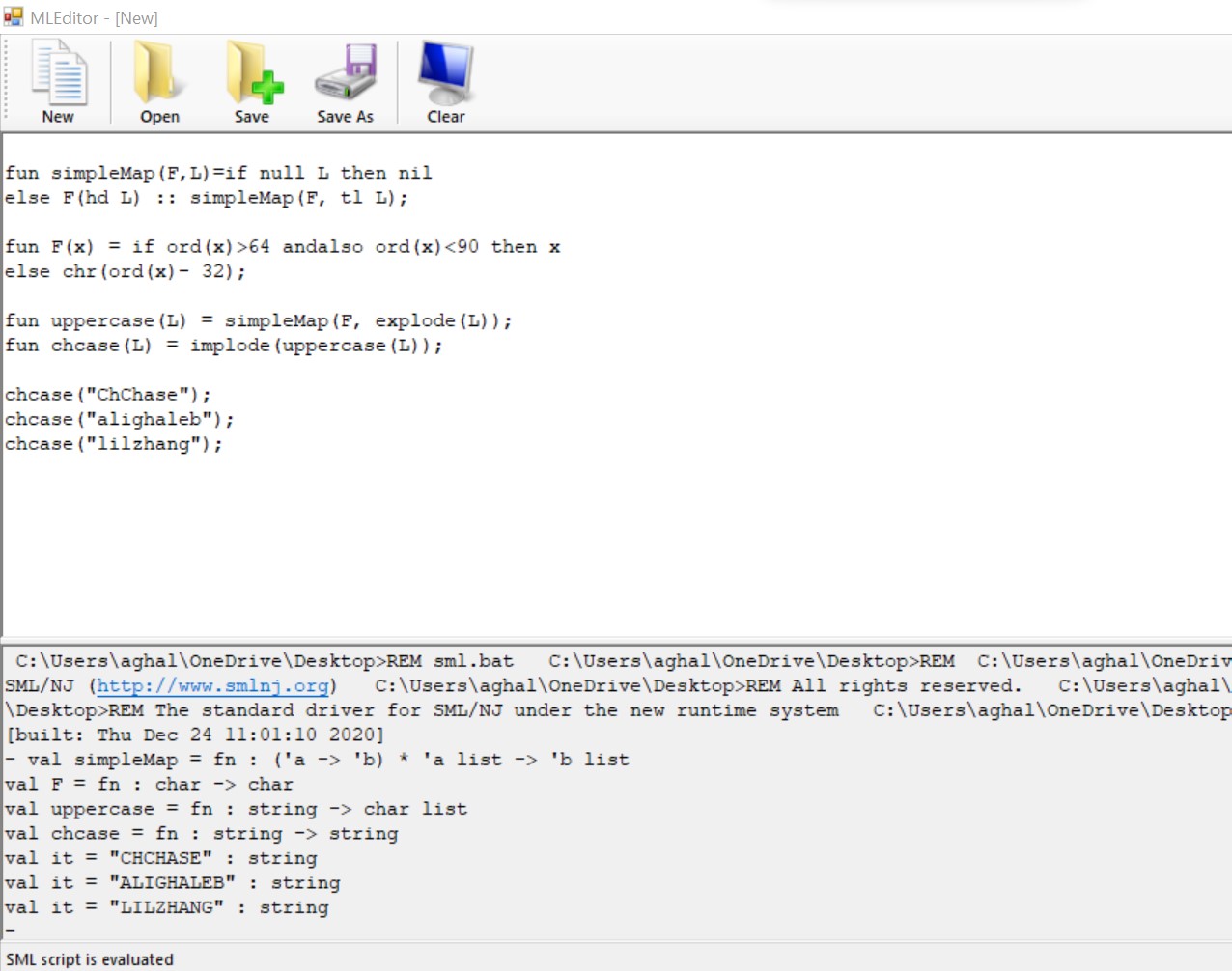
fun chchase(s) = implode(simpleMap(fn x=> if ord(x)>64 andalso ord(x)<90 then x

else chr(ord(x)- 32),explode(s)));

chchase("lilzhang");

chchase("ChCase");

Sample run:



1. 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]];

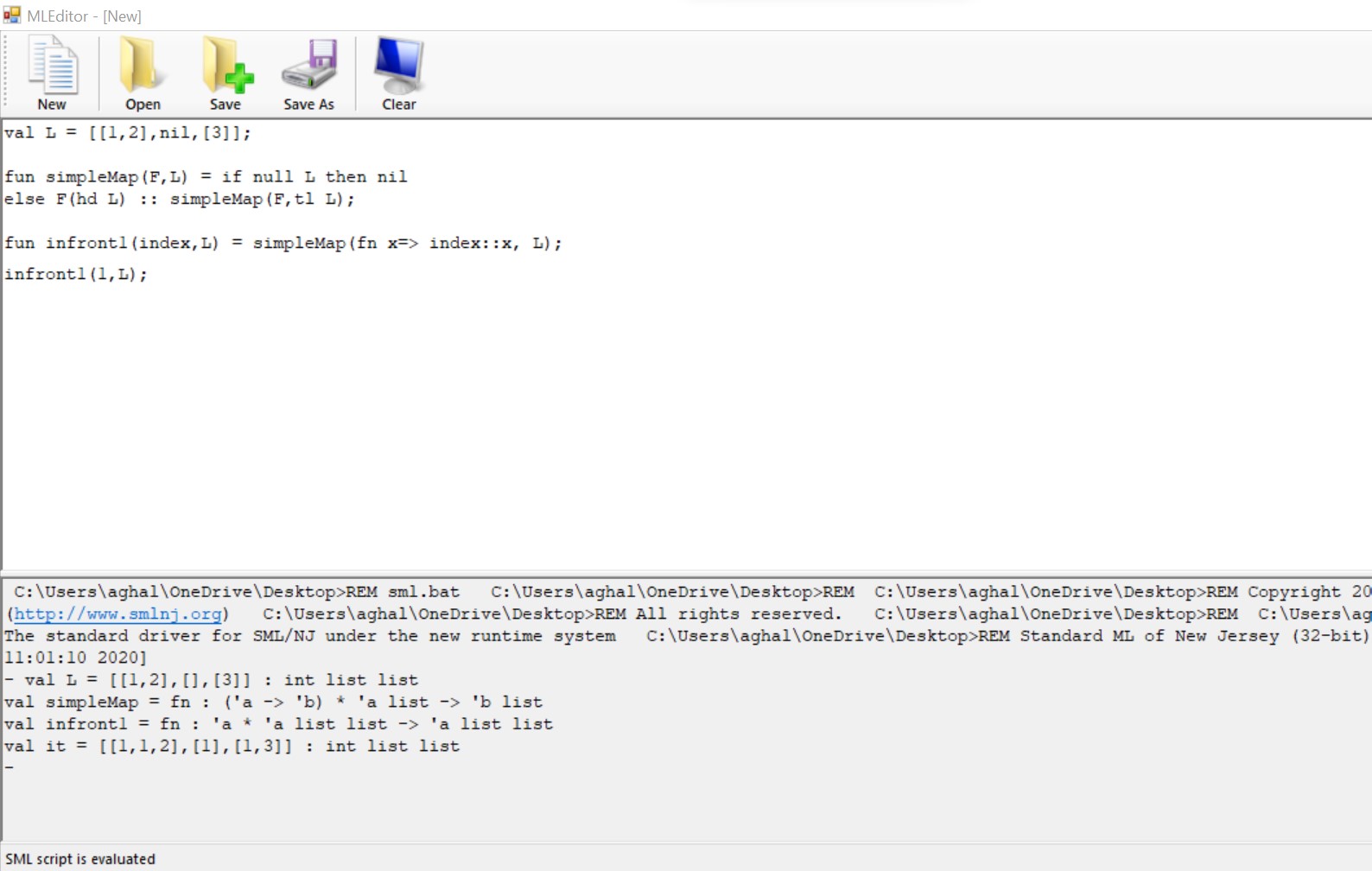
fun simpleMap(F,L) = if null L then nil

else F(hd L) :: simpleMap(F,tl L);

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 add(L,index,value) = if null L andalso index>1 then nil

else if null L andalso index=1 then value::L

else if index=1 then value :: add(L, index+1,value)

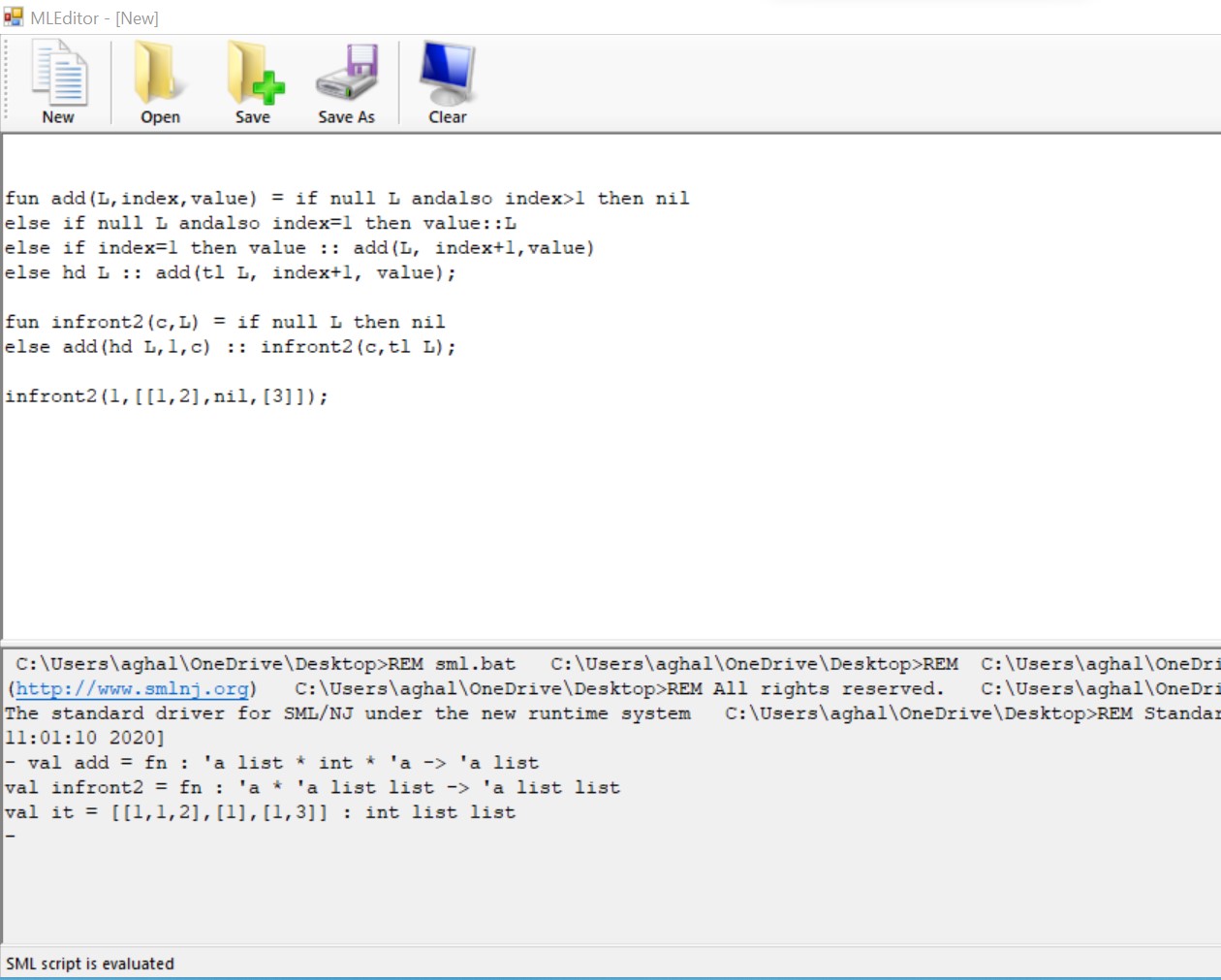
else hd L :: add(tl L, index+1, value);

fun infront2(c,L) = if null L then nil

else add(hd L,1,c) :: infront2(c,tl L);

infront2(1,[[1,2],nil,[3]]);

Sample run:



1. Function name: insea(a,L)

Function description: This function inserts an element to each position of a list recursively.

CODE:

fun append(L1,L2) = if null L1 then L2

else hd L1 :: append(tl L1, L2);

fun insertElement(L1,a,L2) = if null L2 then append(L1,[a]) :: nil

else if null L1 then append([a],L2) :: insertElement([hd L2],a,tl L2)

else append(append(L1,[a]), L2) :: insertElement(append(L1,[hd L2]),a,tl L2);

fun insea(a, L) = insertElement(nil,a, L );

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

insea(0,[1,2,3,4]);

SAMPLE RUN:

