Recursive Evaluator

CSE4102 Project 3, Spring 2018

Sarthak Bhatnagar

3/4/18

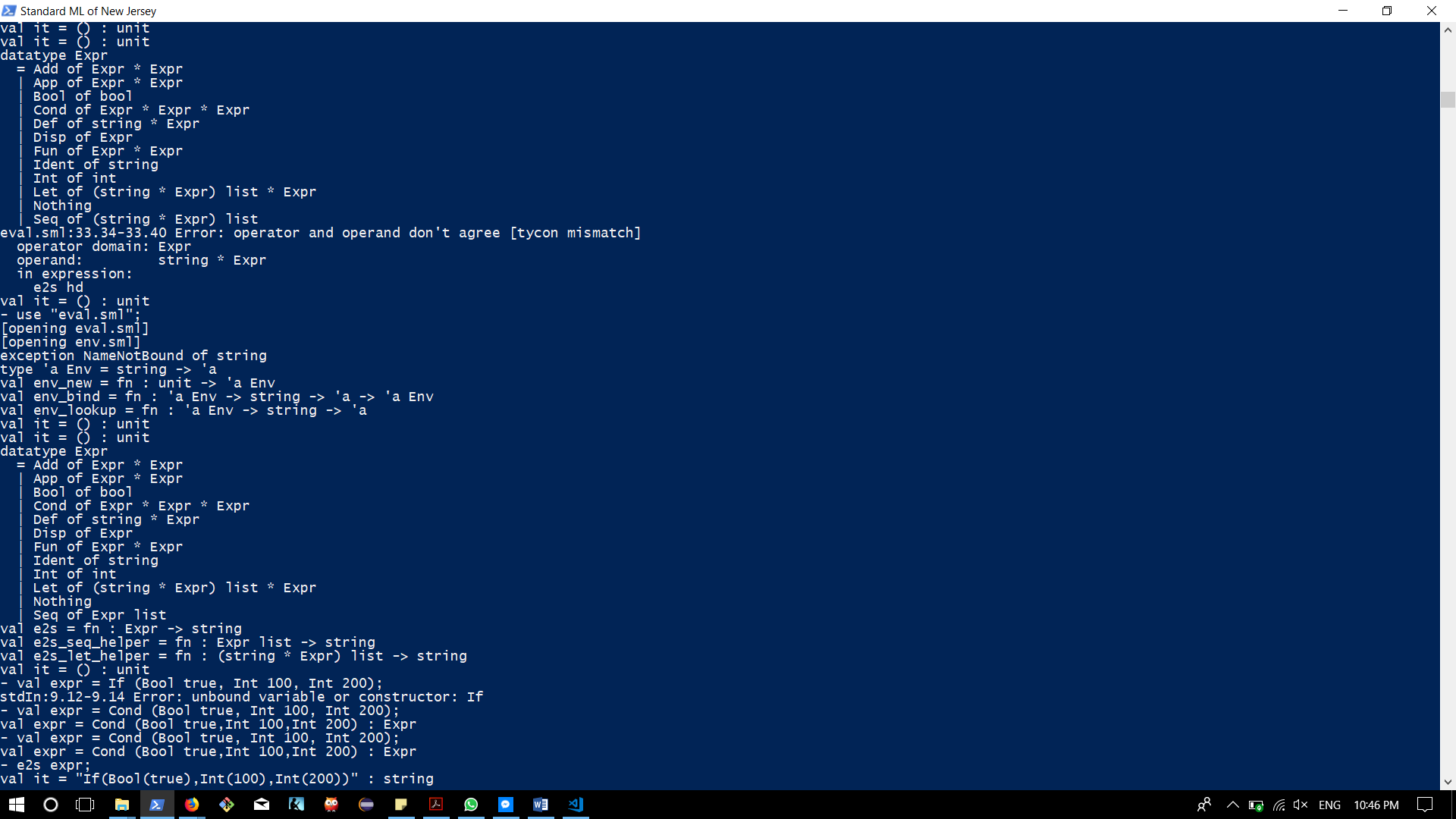
Section: 001

Instructor: Jeffrey A. Meunier

**Introduction**

In this assignment you will write an SML program to evaluate constructs that resemble those from a functional language. This evaluator will be written in a recursive manner: there will be an *eval* function that evaluates expressions and returns their values. For most expressions, the value of the expression depends on the values of the sub-expressions, which will be evaluated recursively.

**Output**



**Source**

use "env.sml";

Control.Print.printDepth := 32;

datatype Expr = Bool of bool

| Int of int

| Add of Expr \* Expr

| Cond of Expr \* Expr \* Expr

| Ident of string

| Let of (string \* Expr) list \* Expr

| Def of (string \* Expr)

| Fun of Expr \* Expr

| App of Expr \* Expr

| Seq of (Expr) list

| Disp of Expr

| Nothing;

fun e2s (Bool b) = "Bool(" ^ (Bool.toString b) ^ ")"

| e2s (Int i) = "Int(" ^ (Int.toString i) ^ ")"

| e2s (Add (x, y)) = "Add(" ^ (e2s x) ^ "," ^ (e2s y) ^ ")"

| e2s (Ident i) = "Ident '" ^ i ^ "'"

| e2s (Disp d) = "Disp (" ^ (e2s d) ^ ")"

| e2s (Cond (cond, conseq, alt)) = "If(" ^ (e2s cond) ^ "," ^ (e2s conseq) ^ "," ^ (e2s alt) ^ ")"

| e2s (Seq s) = "Seq (" ^ (e2s\_seq\_helper s) ^ ")"

| e2s (Def (s,e)) = "Def '" ^ s ^ "', " ^ "(" ^ (e2s e) ^ ")"

| e2s (Fun (e1, e2)) = "Fun (" ^ (e2s e1) ^ "), " ^ (e2s e2)

| e2s (App (e1, e2)) = "App (" ^ (e2s e1) ^ "), " ^ (e2s e2)

| e2s (Let (l, e)) = "Let (" ^ (e2s\_let\_helper l) ^ ") in " ^ (e2s e)

| e2s Nothing = "Nothing"

and e2s\_seq\_helper [] = ""

| e2s\_seq\_helper (hd :: tl) = (e2s hd) ^ "," ^ (e2s\_seq\_helper tl);